**KAUNO TECHNOLOGIJOS UNIVERSITETAS**

**INFORMATIKOS FAKULTETAS**

**OBJEKTINIO PROGRAMAVIMO PROJEKTAVIMAS (Design Patterns) (T120B516)**

**Laboratorinis darbas Nr. 1, 2**

**„Begalinis bėgikas“**

Atliko: IFF – 4/1 gr. Studentas

Audrius Andrijaitis

Miglė Beresinevičiūtė

Matas Balčaitis

Priėmė: Dėst. A. Ušaniov

# Turinys

[Turinys 2](#_Toc499665122)

[Paveikslėlių sąrašas 3](#_Toc499665123)

[Projekto aprašymas 5](#_Toc499665124)

[Funkciniai projekto reikalavimai 5](#_Toc499665125)

[Projekto realizacija 6](#_Toc499665126)

[Projektavimo modelių sąrašas 6](#_Toc499665127)

[Projektavimo modelių pritaikymas 6](#_Toc499665128)

[Adapter pattern 6](#_Toc499665129)

[Strategy pattern 7](#_Toc499665130)

[Factory pattern 9](#_Toc499665131)

[Abstract factory pattern 10](#_Toc499665132)

[Singleton pattern 11](#_Toc499665133)

[Facade pattern 13](#_Toc499665134)

[Decorate pattern 14](#_Toc499665135)

[Bridge pattern 16](#_Toc499665136)

[Template method pattern 19](#_Toc499665137)

[State pattern 22](#_Toc499665138)

[Proxy pattern 24](#_Toc499665139)

[Flyweight pattern 26](#_Toc499665140)

[Chain of responsibility pattern 28](#_Toc499665141)

[Interpreter pattern 30](#_Toc499665142)

[Memento pattern 31](#_Toc499665143)

[Mediator pattern 34](#_Toc499665144)

[Išvados 38](#_Toc499665145)

# Paveikslėlių sąrašas

[1 Pav. Adapter pattern pritaikymas 6](#_Toc499665146)

[2 Pav. Adapter pattern realizacijos kodas 7](#_Toc499665147)

[3 Pav. Adapter pattern realizacijos kodas (bonuses realizacija) 7](#_Toc499665148)

[4 Pav. Strategy pattern pritaikymas 8](#_Toc499665149)

[5 Pav. Strategy pattern realizacijos kodas 8](#_Toc499665150)

[6 Pav. Strategy pattern realizacija (veiksmo realizacija) 8](#_Toc499665151)

[7 Pav. Strategy pattern realizacija (6uolio realizacija) 8](#_Toc499665152)

[8 Pav. Factory pattern pritaikymas 9](#_Toc499665153)

[9 Pav. Factory pattern realizacijos kodas 9](#_Toc499665154)

[10Pav. Factory pattern realizacijos kodas (spalvos realizacija) 10](#_Toc499665155)

[11 Pav. Factory pattern realizacijos kodas (m4lynos spalvos realizacija) 10](#_Toc499665156)

[12 Pav.Abstract factory pattern pritaikymas 11](#_Toc499665157)

[13 Pav. Abstract factory pattern realizacijos kodas 11](#_Toc499665158)

[14 Pav. Sigletop pattern pritaikymas 12](#_Toc499665159)

[15 Pav. Sigleton pattern realizacijos kodas 12](#_Toc499665160)

[16 Pav. Façade pattern pritaikymas 13](#_Toc499665161)

[17 Pav.Facade pattern realizacijos kodas 14](#_Toc499665162)

[18 Pav. Decorate pattern pritaikymas 15](#_Toc499665163)

[19Pav. Decorator pattern realizacijos kodas (žaidėjo realizacija) 15](#_Toc499665164)

[20 Pav. Decorator pattern realizacija (žaidėjo lyties realizacija) 15](#_Toc499665165)

[21 Pav. Decorator pattern realizacija (žaidėjo realizacija) 16](#_Toc499665166)

[22 Pav. Decorator pattern realizacija (greito žaidėjo decorator) 16](#_Toc499665167)

[23 Pav. Bridge pattern pritaikymas 17](#_Toc499665168)

[24 Pav. Bridge pattern realicazija (draw api) 17](#_Toc499665169)

[25 Pav. Bridge pattern realizacija (daiktų realizacija) 17](#_Toc499665170)

[26 Pav. Bridge pattern realizacija (obuolio realizacija) 18](#_Toc499665171)

[27 Pav. Bridge pattern realizacija (teigiamo poveikio daiktų realizacija) 18](#_Toc499665172)

[28 Pav. Template method pattern pritaikymas 19](#_Toc499665173)

[29 Pav. Template method pattern realizacija (MovesContext realizacija) 20](#_Toc499665174)

[30Pav. Template method pattern realizacija (DoubleAction realizacija) 21](#_Toc499665175)

[31 Pav. Template method pattern realizacija (JumpHit realizacija) 21](#_Toc499665176)

[32 Pav.State pattern pritaikymas 22](#_Toc499665177)

[33 Pav. State pattern realizacija (DayTime realizacija) 22](#_Toc499665178)

[34 Pav. State pattern realizacija (BackgroundContet realizacija) 23](#_Toc499665179)

[35 Pav. State pattern realizacija (State metodo realizacija) 23](#_Toc499665180)

[36 Pav. Proxy pattern pritaikymas 24](#_Toc499665181)

[37Pav. Proxy pattern realizacija (Proxy metodo realizacija) 24](#_Toc499665182)

[38Pav. Proxy realizacija (DbSubject realizacija) 25](#_Toc499665183)

[39 Pav. Proxy realizacija (RealDB realizacija) 25](#_Toc499665184)

[40 Pav. Flyweight pattern pritaikymas 26](#_Toc499665185)

[41 Pav. Flyweight realizacija (BackgroundFactory realizacija) 26](#_Toc499665186)

[42 Pav. Flyweight realizacija (Background realizacija) 27](#_Toc499665187)

[43 Pav. Flyweight realizacija (Lake realizacija) 27](#_Toc499665188)

[44 Pav. Chain of responsibility pattern pritiakymas 28](#_Toc499665189)

[45 Pav. Chain of responsibilityt realizacija (HighscoreHandler realizacija) 28](#_Toc499665190)

[46 Pav. Chain of responsibility realizacija (NegativeHandler realizacija) 29](#_Toc499665191)

[47 Pav. Interpreter pattern pritaikymas 30](#_Toc499665192)

[48 Pav. Interpreter realizacija (Counter realizacija) 30](#_Toc499665193)

[49 Pav. Interpreter realizacija (PlusCounter realizacija) 31](#_Toc499665194)

[50 Pav. Interpreter realizacija (NumberCounter realizacija) 31](#_Toc499665195)

[51 Pav. Memento pattern pritaikymas 32](#_Toc499665196)

[52Pav. Memento realizacija (GameTaker realizacija) 32](#_Toc499665197)

[53 Pav. Memento realizacija 33](#_Toc499665198)

[54 Pav. Mediator pattern pritaikymas 34](#_Toc499665199)

[55 Pav. Mediator realizacija 34](#_Toc499665200)

[56 Pav. Mediator realizacija (Interlocutor realizacija) 34](#_Toc499665201)

[57 Pav. Mediator realizacija (ConcreteMediator realizacija) 35](#_Toc499665202)

[58Pav. Mediator realizacija (User realizacija) 36](#_Toc499665203)

[59 Pav. Mediator realizacija (Bot realizacija) 37](#_Toc499665204)

# Projekto aprašymas

Mūsų komanda siekdama įgyvendinti projektą bei išpildyti visus reikalavimus nusprendė kurti žaidimo „Endless Runner“ analogą. Pasirinkome būtent šį žaidimą dėl jo lengvo iš plečiamumo bei lengvai perkandamos pagrindinės žaidimo idėjos, kuri patiko visiems komandos nariams.

Mūsų tikslas sukurti „Endless Runner“ žaidimo varikliuką nerealizuojant grafinės sąsajos – žaidimo tęstinumą bei įvykius bus galima stebėti per CLI (valdymas vyks per komandinę eilutę).

Mūsų tikslas įgyvendinti pagrindinį žaidimo funkcionalumą bei pridėti kelis naujus patobulinimus (objektų rinkimas, kliūčių griovimas).

# Funkciniai projekto reikalavimai

1. Žaidėjo gyvavimo trukmė begalinė arba iki pirmo susidūrimo su kliūtimi;
2. Žaidėjas gali rinkti objektus, kurie jam suteikia papildomų galių arba jas sumažina;
3. Žaidėjas gali atlikti tokius veiksmus kaip šokti, tūpti bei trenkti;
4. Žemėlapio generavimas automatizuotas – objektų pozicijos atsitiktinės, tačiau su realizuotomis apsaugomis (turi egzistuoti minimalus atstumas tarp objektų);
5. Papildomų objektų generavimas automatizuotas.

# Projekto realizacija

## Projektavimo modelių sąrašas

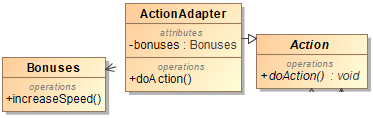
Projekte panaudojome šiuos projektavimo modelius:

* Elgsenos (behavior):
  + Strategy pattern
  + Template method
  + State
  + Chain of responsibility
  + Interpreter
  + Mediator
  + Memento
* Kūrimo (creational):
  + Singleton pattern
  + Factory pattern
  + Abstract factory pattern
* Struktūriniai (structural):
  + Facade pattern
  + Decorator pattern
  + Bridge
  + Proxy
  + Flyweight

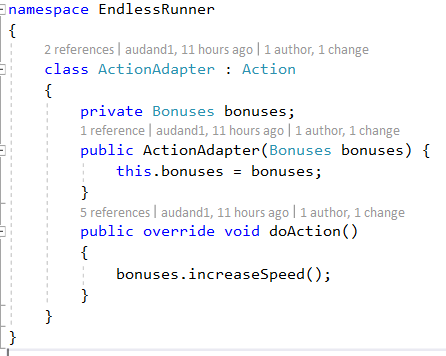
## Projektavimo modelių pritaikymas

### Adapter pattern

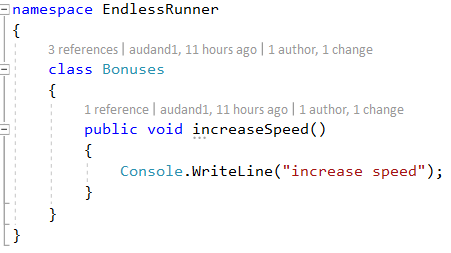
Šį projektavimo modelį pritaikėme apjungti žaidėjo galimybę sustiprinti savo savybes (greitį) su objektais, kurių paėmimas gali atitinkamai sustiprinti ar susilpninti jas. Žemėlapyje atsitiktinai atsiranda obuoliai ir persikai (objektai), juos „paėmus“ atitinkamai paveikiama žaidėjo greitis. Poveikio stiprumas priklauso nuo objekto spalvos. Žemiau pateiktame paveikslėlyje matyti kaip panaudojant adapter pattern apjungti šie objektai su poveikio metodais.



Pav. Adapter pattern pritaikymas



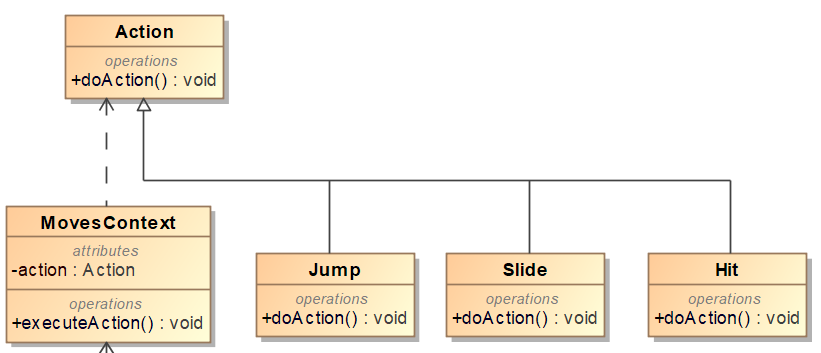
Pav. Adapter pattern realizacijos kodas



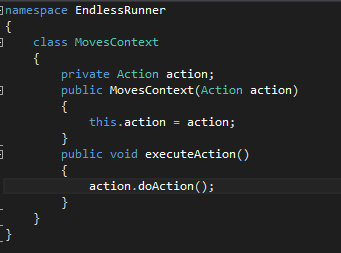
Pav. Adapter pattern realizacijos kodas (bonuses realizacija)

### Strategy pattern

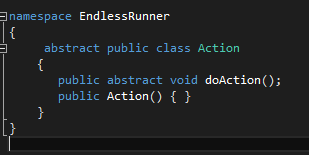
Šį projektavimo modelį pritaikėme žaidėjo veiksmų (šokti, čiuožti, trenkti) realizacijai. Pasirinkome būtent šį projektavimo modelį, kadangi, mūsų nuomone, jis tinkamiausias iš elgsenos modulių mūsų projektui ir geriausiai atspindi mūsų norimo funkcionalumo realizaciją. Žemiau pateiktame paveikslėlyje matyti kaip panaudojant stategy pattern realizuoti veiksmai, kurios gali atlikti žaidėjas.



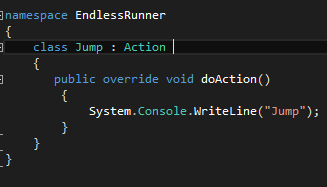
Pav. Strategy pattern pritaikymas



Pav. Strategy pattern realizacijos kodas



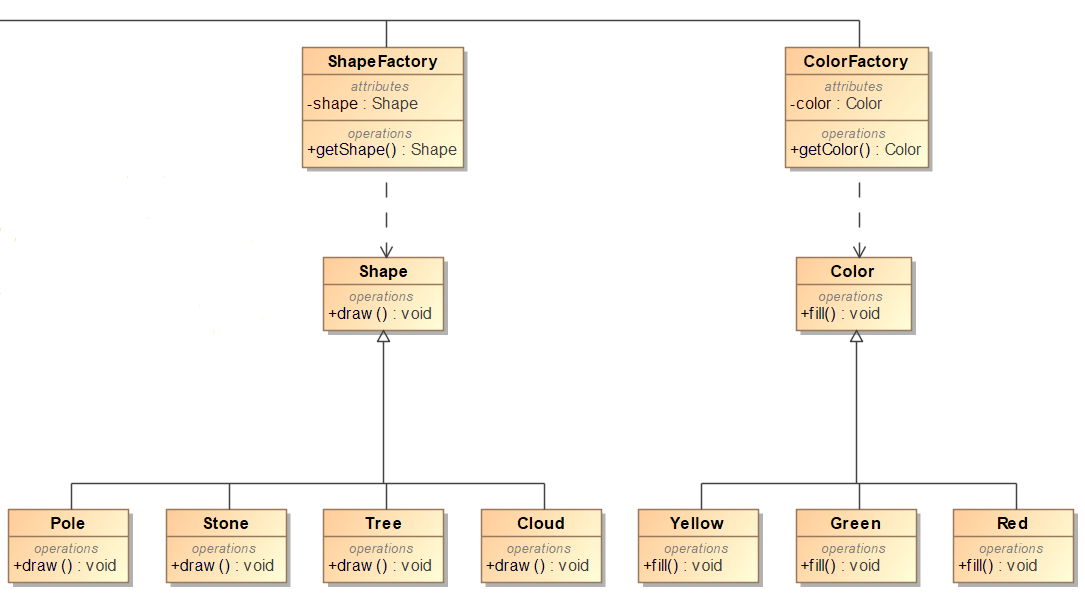
Pav. Strategy pattern realizacija (veiksmo realizacija)



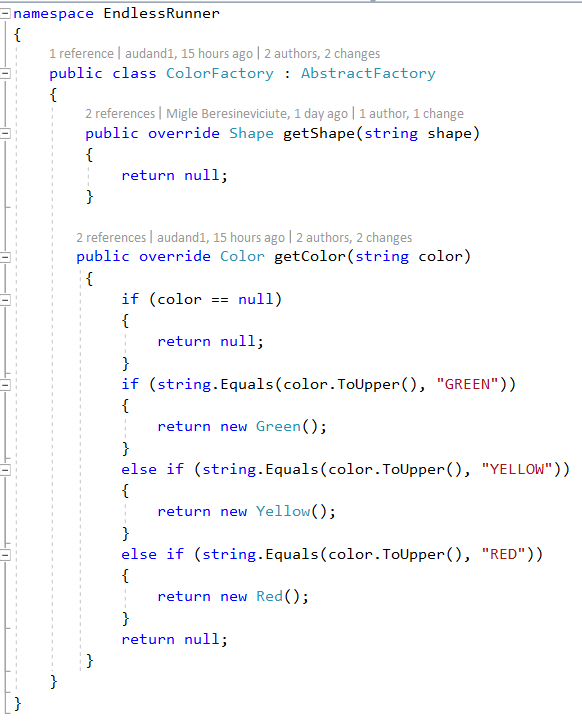
Pav. Strategy pattern realizacija (6uolio realizacija)

### Factory pattern

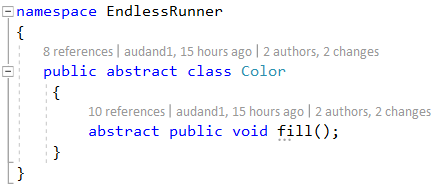
Šį projektavimo modelį pritaikėme objektų (kliūčių žemėlapyje) bei jų spalvų kūrimui. Pasirinkome būtent šį iš kūrimo modelių, nes tai vienas populiariausių šio tipo modelių bei atitinka mūsų tikslą – galimybę, kurti objektus, kurių kūrimo logika neprieinama vartotojui. Žemiau pateiktame paveikslėlyje matyti kaip realizuotas factory pattern žemėlapio objektų kūrimui bei jų spalvų kūrimui.



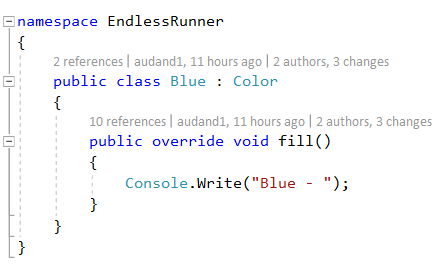
Pav. Factory pattern pritaikymas



Pav. Factory pattern realizacijos kodas



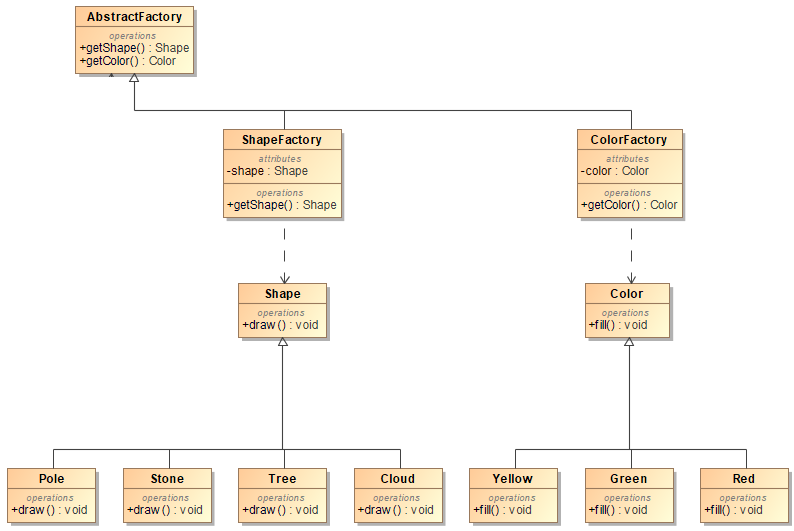
Pav. Factory pattern realizacijos kodas (spalvos realizacija)



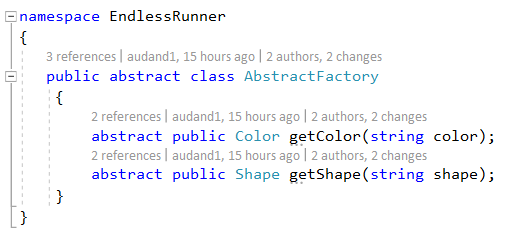
Pav. Factory pattern realizacijos kodas (m4lynos spalvos realizacija)

### Abstract factory pattern

Šį projektavimo modelį pritaikėme jau anksčiau minėtų factory pattern panaudojimo atveju apjungimui. Pasirinkome būtent šį kūrimo modelį, nes jo paskirtis ir yra apjungti keliems factory patterns. Žemiau pateiktame paveikslėlyje matyti kaip du fatory patterns (žemėlapio objektų bei spalvų) apjungiami naudojant abstract factory



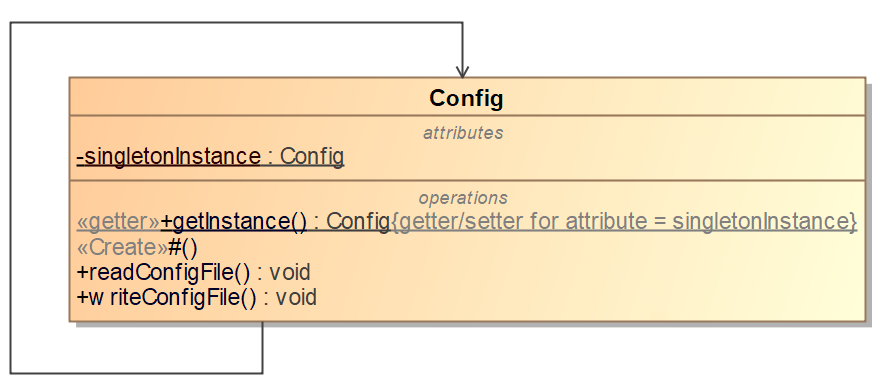
Pav.Abstract factory pattern pritaikymas



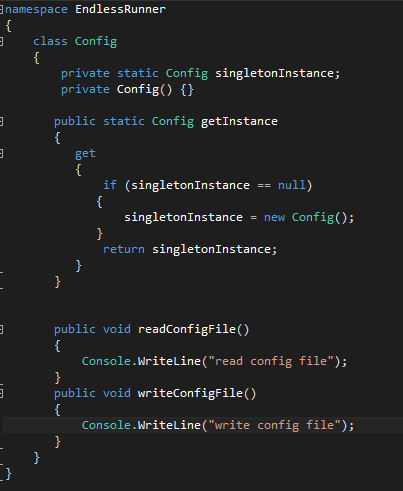
Pav. Abstract factory pattern realizacijos kodas

### Singleton pattern

Šį projektavimo modelį pritaikėme konfigūracijų failo kūrimui. Singleton projektavimo modelis yra vienas paprasčiausių kūrimo modelių skirtų sukurti objektui, kuriam reikia užtikrinti, jog jis bus toks vienintelis. Būtent todėl ir pasirinkome šį projektavimo modelį konfigūracijų failo kūrimui (projekte reikalingas tik vienas konfigūracijų failas). Žemiau pateiktame paveikslėlyje matyti kaip pritaikėme singleton projektavimo modelį savo konfigūracijų failo kūrimui.



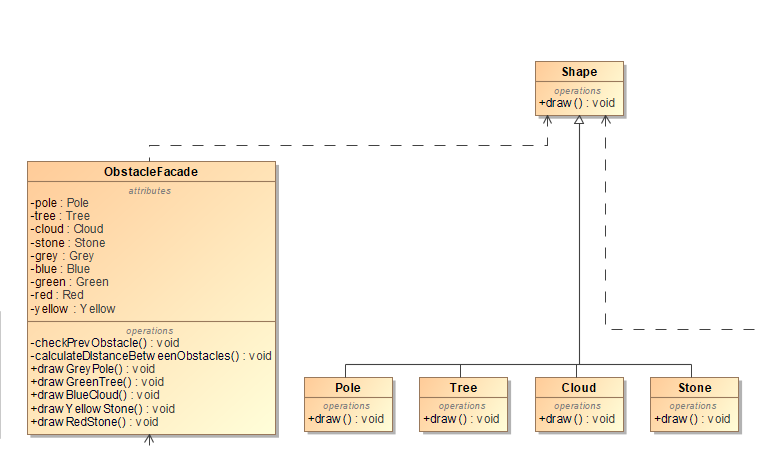
Pav. Sigletop pattern pritaikymas



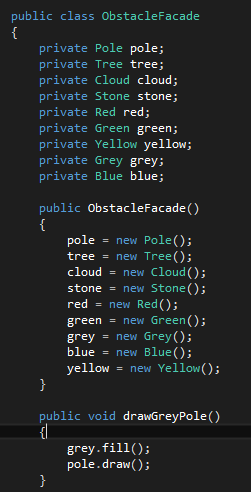
Pav. Sigleton pattern realizacijos kodas

### Facade pattern

Šį projektavimo modelį pritaikėme kliūčių žemėlapyje atsiradimo logikai realizuoti. Mūsų nuomone, façade struktūrinis projektavimo modelis labiausiai atitiko mūsų poreikius, t.y. suteikia galimybę sudėtinga posistemę apibendrinti į vieną bendrą paprastą sąsaja. Tai suteikia mums galimybę paslėpti sudėtinga kliūčių generavimo logiką bei taip pagerinti projekto suprantamumą bei skaitomumą. Žemiau pateiktame paveikslėlyje matyti kaip pritaikėme façade projektavimo modelį kliūčių generavimo logikai apibendrinti.



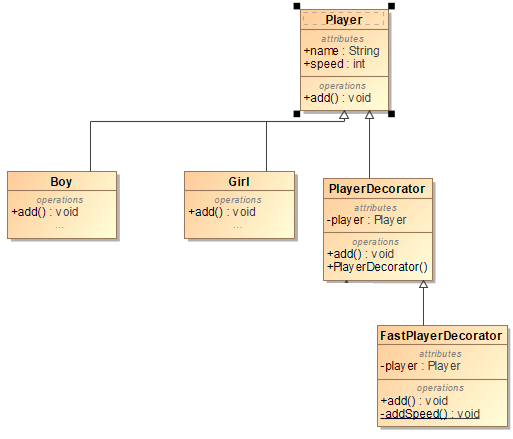
Pav. Façade pattern pritaikymas



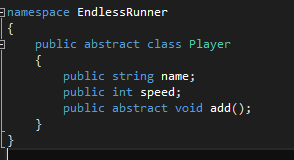
Pav.Facade pattern realizacijos kodas

### Decorate pattern

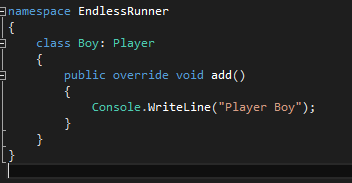
Šį projektavimo modelį pritaikėme žaidėjo savybių (jėgos bei greičio) keitimui. Decorate pattern priklauso struktūrinių projektavimo modelių šeimai bei yra skirtas pridėti naują funkcionalumą jau egzistuojančiam objektui nepažeidžiant jo struktūros. Kadangi tai atitiko visus mūsų išsikeltus reikalavimus šiai posistemei, nusprendėme, kad decorate pattern yra būtent tai ko mums reikia. Žemiau pateiktame paveikslėlyje galima matyti decorate pattern pritaikymą žaidėjo savybių pridėjimui.



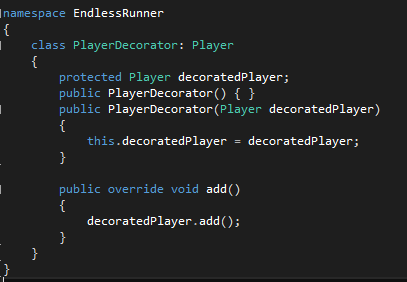
Pav. Decorate pattern pritaikymas



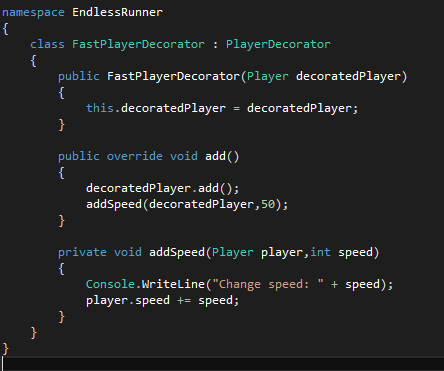
Pav. Decorator pattern realizacijos kodas (žaidėjo realizacija)



Pav. Decorator pattern realizacija (žaidėjo lyties realizacija)



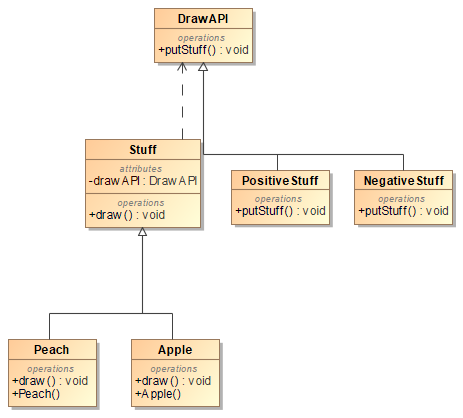
Pav. Decorator pattern realizacija (žaidėjo realizacija)



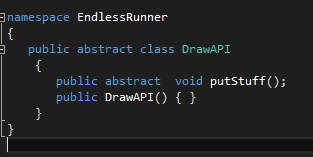
Pav. Decorator pattern realizacija (greito žaidėjo decorator)

### Bridge pattern

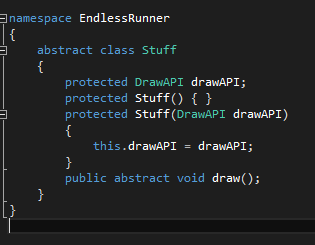
Šį projektavimo modelį pritaikėme objektų, kuriuos gali rinkti žaidėjas kūrimui. Bridge projektavimo modelis taip pat priklauso struktūrinių projektavimo modelių šeimai ir yra skirtas implementacijos atskyrimui nuo abstrakcijos, kad abi dalys galėtų gyvuoti nepriklausomai. Būtent todėl nusprendėme, kad jis mums yra tinkamiausias šioje situacijoje.



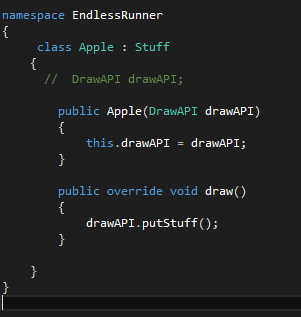
Pav. Bridge pattern pritaikymas



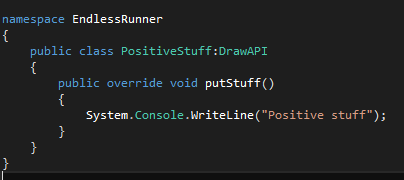
Pav. Bridge pattern realicazija (draw api)



Pav. Bridge pattern realizacija (daiktų realizacija)



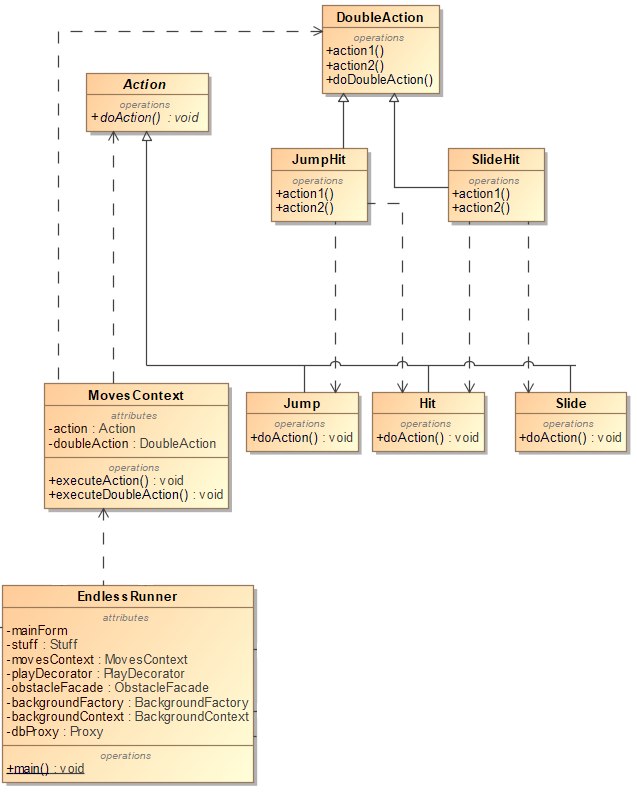
Pav. Bridge pattern realizacija (obuolio realizacija)



Pav. Bridge pattern realizacija (teigiamo poveikio daiktų realizacija)

### Template method pattern

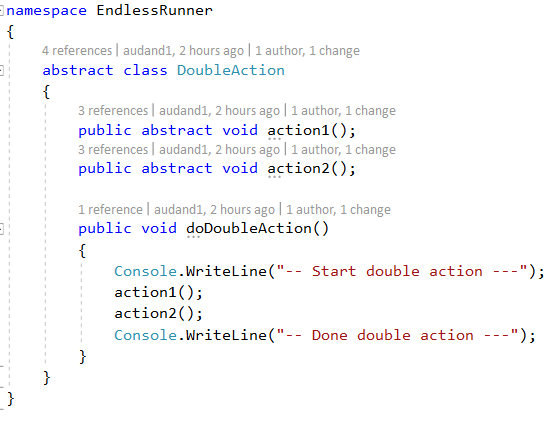
Šį projektavimo modelį pritaikėme kombinuotų žaidėjo veiksmų (šokti ir trenkti, čiuožti ir trenkti) realizacijai. Pasirinkome būtent šį projektavimo modelį, kadangi, mūsų nuomone, jis tinkamiausias iš elgsenos modulių mūsų projektui ir geriausiai atspindi mūsų norimo funkcionalumo realizaciją. Žemiau pateiktame paveikslėlyje matyti kaip panaudojant template method pattern realizuoti kombinuoti veiksmai, kurios gali atlikti žaidėjas.



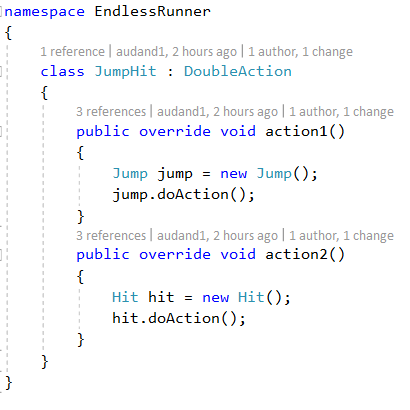
Pav. Template method pattern pritaikymas

#### 

Pav. Template method pattern realizacija (MovesContext realizacija)



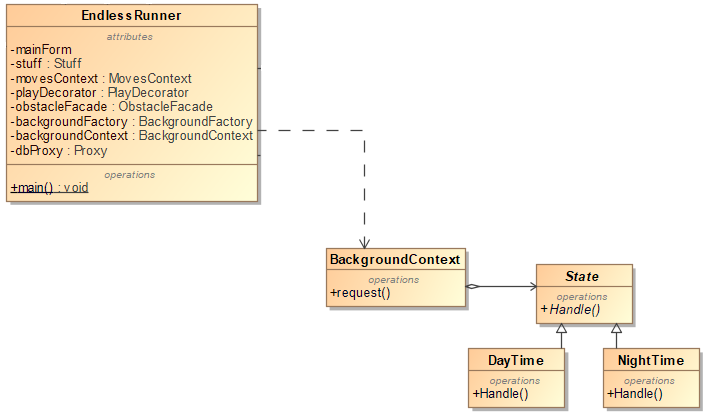
Pav. Template method pattern realizacija (DoubleAction realizacija)



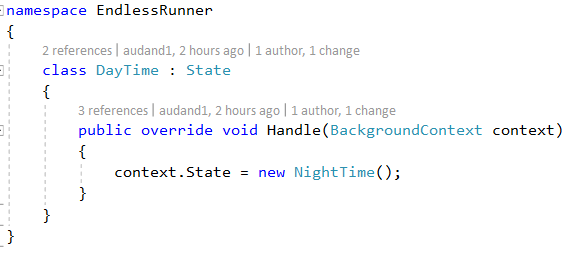
Pav. Template method pattern realizacija (JumpHit realizacija)

### State pattern

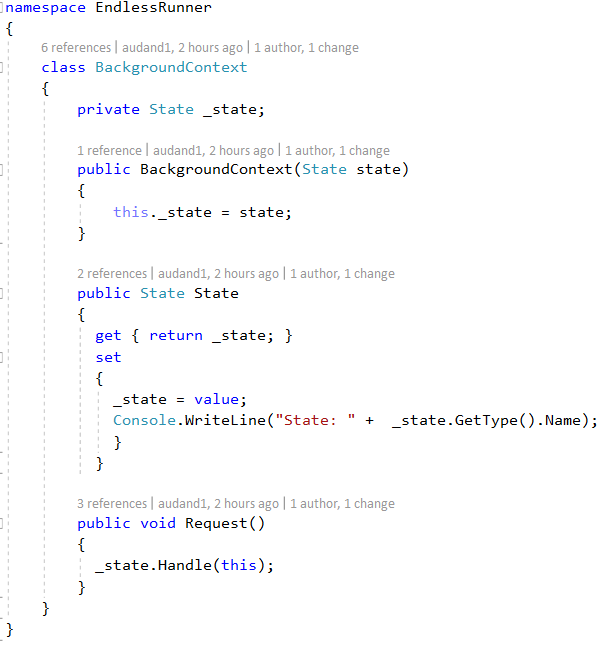
Šį projektavimo modelį pritaikėme paros meto nustatymo (diena arba naktis) realizacijai. Pasirinkome būtent šį projektavimo modelį, kadangi, mūsų nuomone, jis tinkamiausias iš elgsenos modulių šios projekto dalies realizacijai ir geriausiai atspindi mūsų norimo funkcionalumo realizaciją. Žemiau pateiktame paveikslėlyje matyti kaip panaudojant state pattern realizuoti paros meto nustatymui.



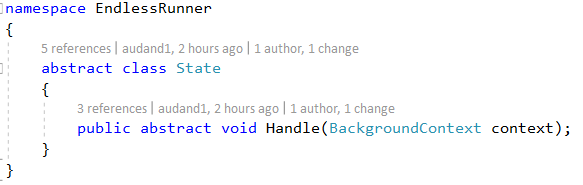
Pav.State pattern pritaikymas



Pav. State pattern realizacija (DayTime realizacija)



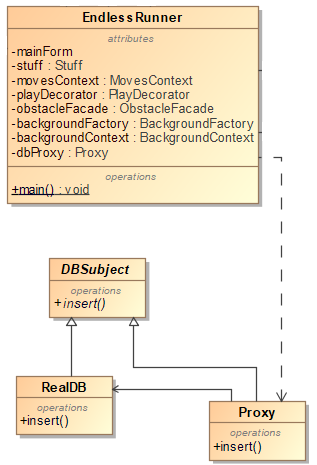
Pav. State pattern realizacija (BackgroundContet realizacija)



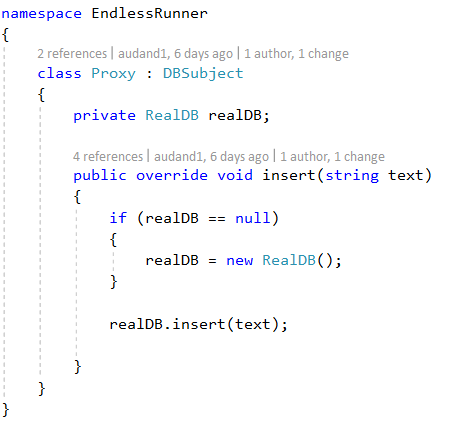
Pav. State pattern realizacija (State metodo realizacija)

### Proxy pattern

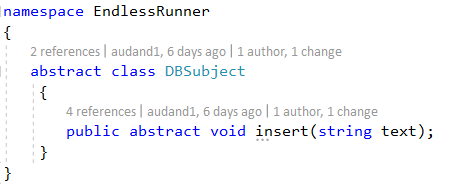
Šį projektavimo modelį pritaikėme duomenų į duombazę įrašymui. Mūsų nuomone, struktūrinis projektavimo modelis proxy labiausiai atitiko mūsų poreikius, t.y. galimybė bendrauti su duomenų baze išvengiant tiesioginių kreipinių iš kliento. Žemiau pateiktame paveikslėlyje matyti kaip pritaikėme proxy projektavimo modelį bendravimo su duomenų baze realizacijai.



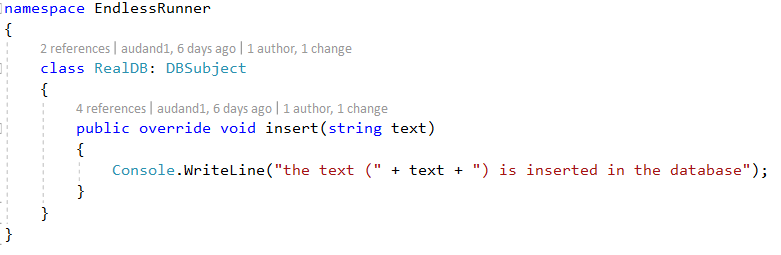
Pav. Proxy pattern pritaikymas



Pav. Proxy pattern realizacija (Proxy metodo realizacija)



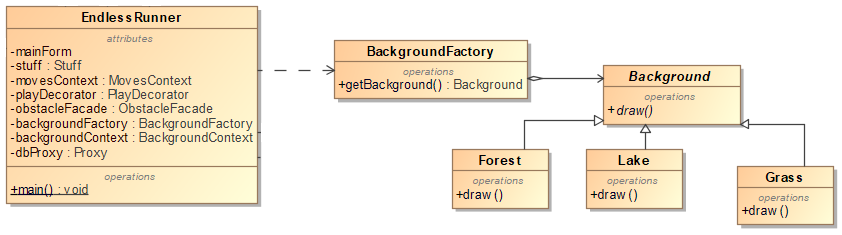
Pav. Proxy realizacija (DbSubject realizacija)



Pav. Proxy realizacija (RealDB realizacija)

### Flyweight pattern

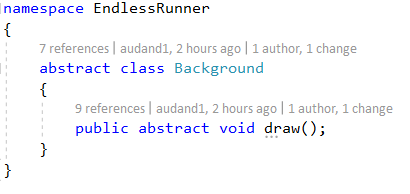
Šį projektavimo modelį pritaikėme antrame plane esančių objektų kūrimui. Mūsų nuomone, struktūrinis projektavimo modelis flyweight labiausiai atitiko mūsų poreikius, t.y. galimybė kurti daug vienodų objektų tausojant atmintį. Žemiau pateiktame paveikslėlyje matyti kaip pritaikėme flyweight projektavimo modelį antraplanių objektų kūrimui.



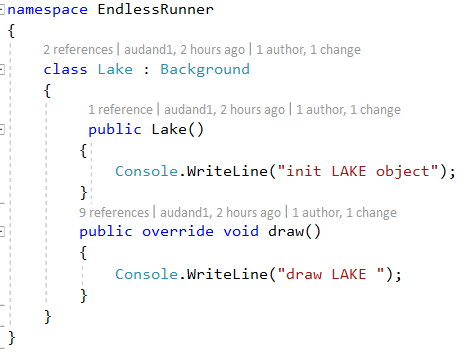
Pav. Flyweight pattern pritaikymas



Pav. Flyweight realizacija (BackgroundFactory realizacija)



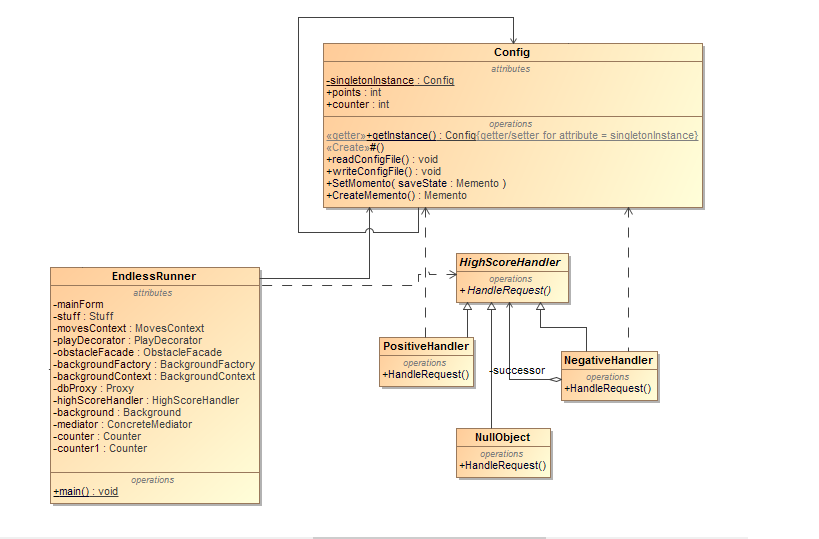
Pav. Flyweight realizacija (Background realizacija)



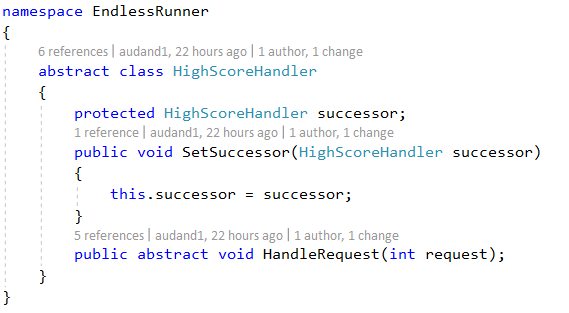
Pav. Flyweight realizacija (Lake realizacija)

### Chain of responsibility pattern

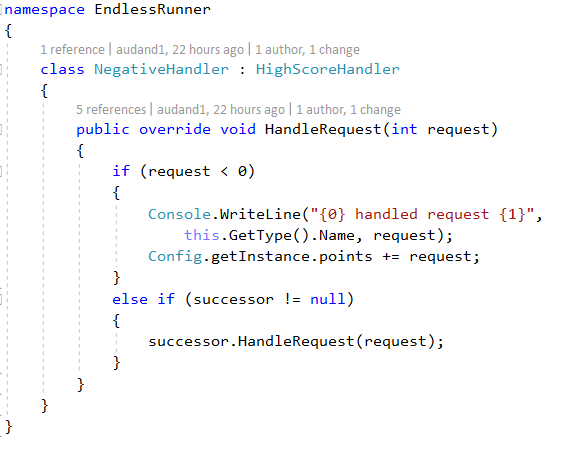
Šį projektavimo modelį pritaikėme žaidėjo surinktų taškų skaičiavimui. Pasirinkome būtent šį projektavimo modelį, kadangi, mūsų nuomone, jis tinkamiausias iš elgsenos modulių „highscores“ skaičiavimui. Žemiau pateiktame paveikslėlyje matyti kaip panaudojant chain of responsibility pattern realizuoti paros meto nustatymui.



Pav. Chain of responsibility pattern pritiakymas



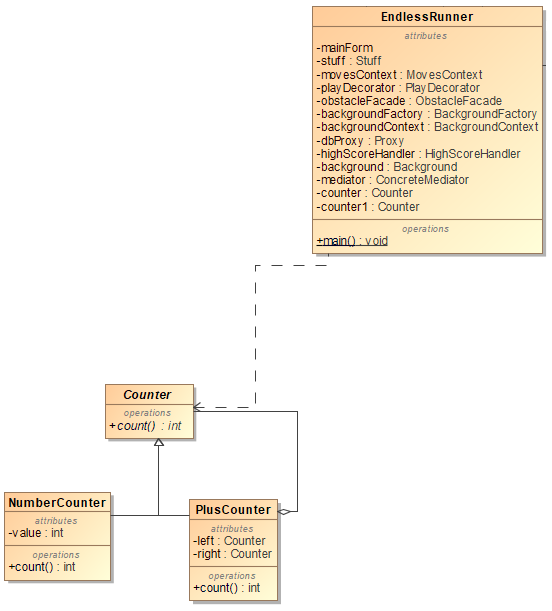
Pav. Chain of responsibilityt realizacija (HighscoreHandler realizacija)



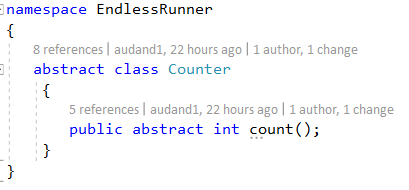
Pav. Chain of responsibility realizacija (NegativeHandler realizacija)

### Interpreter pattern

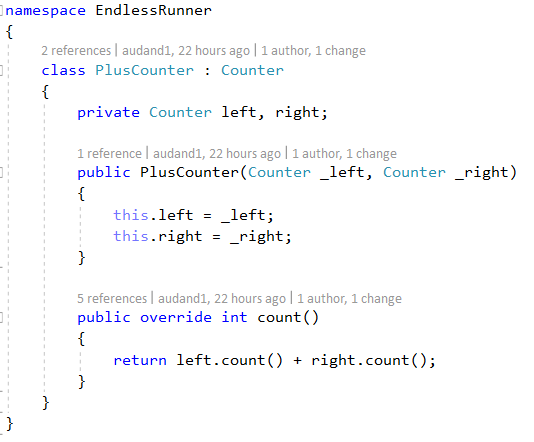
Šį projektavimo modelį pritaikėme „rekordų“ (highscores) skaičiavimui. Pasirinkome būtent šį projektavimo modelį, kadangi, mūsų nuomone, jis tinkamiausias iš elgsenos modulių „highscores“. Žemiau pateiktame paveikslėlyje matyti kaip panaudojant chain of responsibility pattern realizuoti paros meto nustatymui.



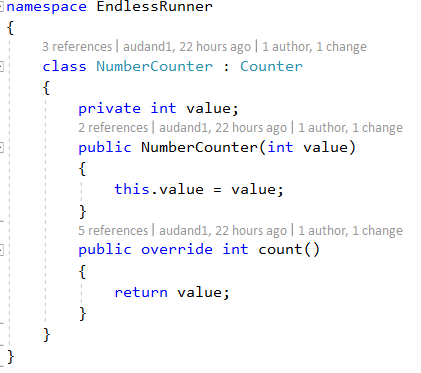
Pav. Interpreter pattern pritaikymas



Pav. Interpreter realizacija (Counter realizacija)



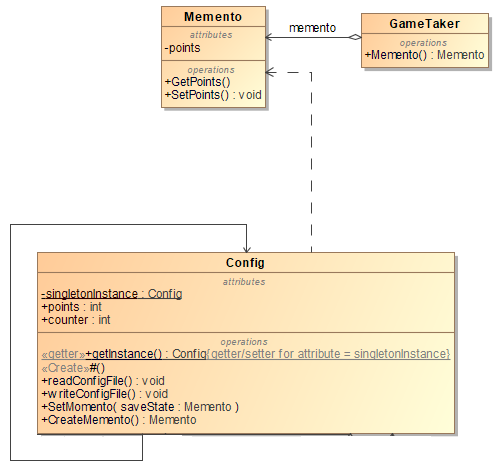
Pav. Interpreter realizacija (PlusCounter realizacija)



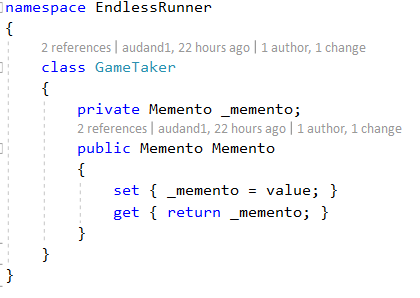
Pav. Interpreter realizacija (NumberCounter realizacija)

### Memento pattern

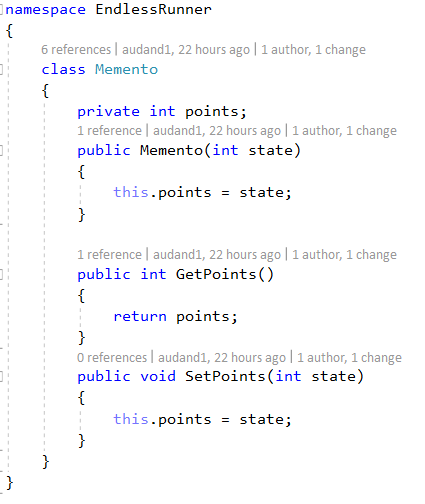
Šį projektavimo modelį pritaikėme taškų išsaugojimui. Pasirinkome būtent šį projektavimo modelį, kadangi, mūsų nuomone, jis tinkamiausias iš elgsenos modulių mūsų užduoties realizacijai. Pagrindinis tikslas – išsaugoti turimus taškus, tam, kad žaidėjui panaudojus antrą gyvybę ar pan. būtų galima testi žaidimą toliau, o ne pradėti vėl nuo nulio. Žemiau pateiktame paveikslėlyje matyti kaip panaudojant memento pattern realizuoti taškų kiekio išsaugojimui.



Pav. Memento pattern pritaikymas



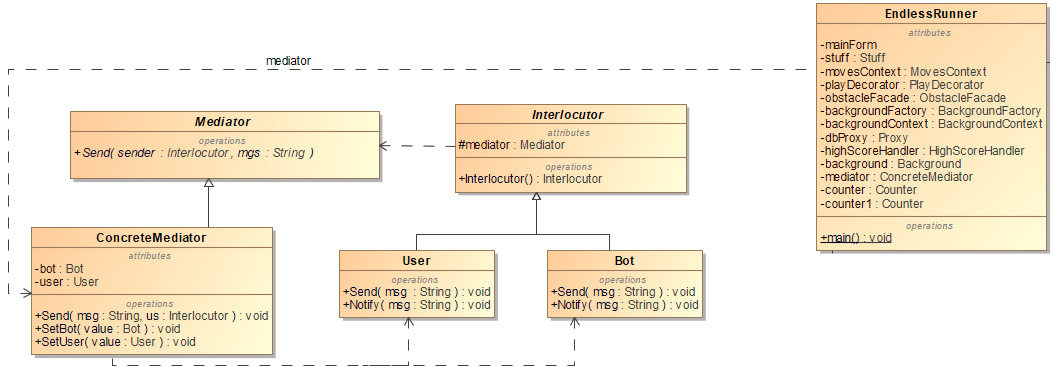
Pav. Memento realizacija (GameTaker realizacija)



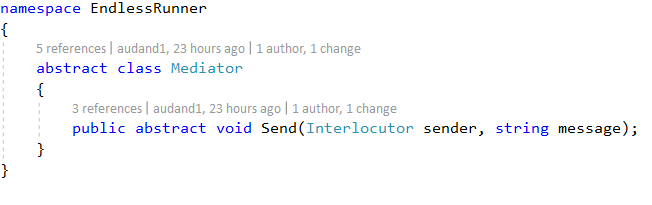
Pav. Memento realizacija

### Mediator pattern

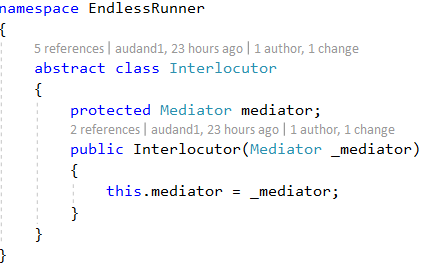
Šį projektavimo modelį pritaikėme „rekordų“ (highscores) skaičiavimui. Pasirinkome būtent šį projektavimo modelį, kadangi, mūsų nuomone, jis tinkamiausias iš elgsenos modulių „highscores“. Žemiau pateiktame paveikslėlyje matyti kaip panaudojant chain of responsibility pattern realizuoti paros meto nustatymui.



Pav. Mediator pattern pritaikymas



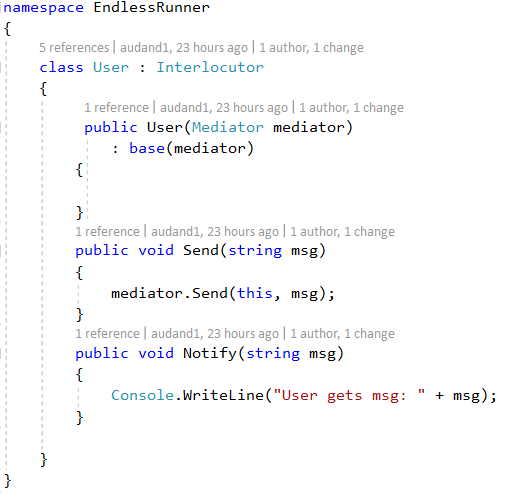
Pav. Mediator realizacija



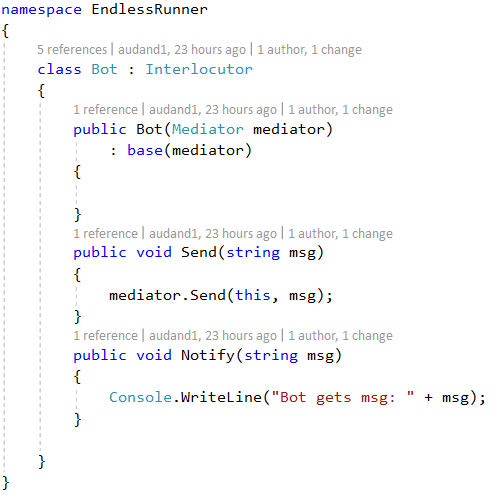
Pav. Mediator realizacija (Interlocutor realizacija)



Pav. Mediator realizacija (ConcreteMediator realizacija)



Pav. Mediator realizacija (User realizacija)



Pav. Mediator realizacija (Bot realizacija)

# Išvados

Panaudoję įvairius elgsenos, kūrimo bei struktūrinius projektavimo modelius pastebėjome jų teikiamą naudą, kuomet projektas tampa lengviau suprantamas, o svarbiausia lengviau koreguojamas.