

**KAUNO TECHNOLOGIJOS UNIVERSITETAS**

**INFORMATIKOS FAKULTETAS**

# TAIKOMOSIOS INFORMATIKOS KATEDRA

OBJEKTINIS PROGRAMŲ PROJEKTAVIMAS (T120B516)

Laboratorinis darbas Nr. 1

### Atliko:

Paulius Ratkevičius IFF 8/12 Dovydas Bražas IFF-8/6 Antanas Stepanauskas IFF-8/6

### Priėmė:

Andrej Ušaniov

### KAUNAS, 2021

# TURINYS

Contents

[TURINYS 1](#_Toc90641604)

[1. PROJEKTO APRAŠYMAS 6](#_Toc90641605)

[1.1 pav. Žaidimo prototipo nuotrauka 6](#_Toc90641606)

[2. ŽAIDIMO REIKALAVIMAI 7](#_Toc90641607)

[2.1. ŽAIDIMO LYGIAI 7](#_Toc90641608)

[2.2. PIRMASIS LYGIS 7](#_Toc90641609)

[2.3. ANTRAS LYGIS 7](#_Toc90641610)

[2.4. TREČIAS LYGIS 7](#_Toc90641611)

[3. PROJEKTUI NAUDOTOS TECHNOLOGIJOS 8](#_Toc90641612)

[4. USE CASE DIAGRAMA 8](#_Toc90641613)

[4.2 pav. Use case diagrama 8](#_Toc90641614)

[5. KLASIŲ DIAGRAMA 9](#_Toc90641615)

[5.3 pav. Klasių diagrama 9](#_Toc90641616)

[5.1. ŠABLONAI 10](#_Toc90641617)

[5.1.1. Singleton 10](#_Toc90641618)

[5.4 pav. Singleton diagrama 10](#_Toc90641619)

[5.1.2. Strategy 21](#_Toc90641620)

[5.5 pav. Strategy diagrama 21](#_Toc90641621)

[5.1.3. Decorator 32](#_Toc90641622)

[5.6 pav. Decorator diagrama 32](#_Toc90641623)

[5.1.4. Abstract Factory 43](#_Toc90641624)

[5.7 pav. Abstract Factory diagrama 43](#_Toc90641625)

[5.1.5. Observer 66](#_Toc90641626)

[5.8 pav. Observer diagrama 66](#_Toc90641627)

[5.1.6. Facade 81](#_Toc90641628)

[5.10 pav. Facade diagrama 81](#_Toc90641629)

[5.1.7. Factory 107](#_Toc90641630)

[5.11 pav. Factory diagrama 107](#_Toc90641631)

[5.12 pav. Builder diagrama 111](#_Toc90641633)

[5.13 pav. Prototype diagrama 123](#_Toc90641635)

[6. PROJEKTO APRAŠYMAS ANTRA DALIS 124](#_Toc90641636)

[6.1 pav. Žaidimo prototipo nuotrauka 124](#_Toc90641637)

[7. ŽAIDIMO REIKALAVIMAI 125](#_Toc90641638)

[ŽAIDIMO LYGIAI 125](#_Toc90641639)

[PIRMASIS LYGIS 125](#_Toc90641640)

[ANTRAS LYGIS 125](#_Toc90641641)

[TREČIAS LYGIS 125](#_Toc90641642)

[8. PROJEKTUI NAUDOTOS TECHNOLOGIJOS 126](#_Toc90641643)

[9. USE CASE DIAGRAMA 126](#_Toc90641644)

[4.2 pav. Use case diagrama 126](#_Toc90641645)

[10. KLASIŲ DIAGRAMA 127](#_Toc90641646)

[5.14 pav. Klasių diagrama 127](#_Toc90641647)

[11 ŠABLONAI 128](#_Toc90641648)

[GameState 128](#_Toc90641649)

[GameState šablono prireikė norint įdėti į žaidimą skirtingas būsenas. Tai yra : 128](#_Toc90641650)

[ Žaidimo laukimas 128](#_Toc90641651)

[ Žaidimo vykdymas 128](#_Toc90641652)

[ Žaidimo užbaigimo bandymas 128](#_Toc90641653)

[ Žaidimo lentos pakeitimas į kitą 128](#_Toc90641654)

[Šis šablonas puikiai padėjo įgyvendinti šią idėją. 128](#_Toc90641655)

[UML diagrama: 128](#_Toc90641656)

[128](#_Toc90641657)

[Kodas: 129](#_Toc90641658)

[Proxy šablonas 165](#_Toc90641659)

[166](#_Toc90641660)

[Memento 190](#_Toc90641661)

[Kilo idėja į žaidimą įdėti įgūdį, kuris leidžia vartotojui “grįžti laiku” ir nekreipti dėmesio į klaidas, kurias padarė. Šiai idėjai tikslingai pravertė memento šablonas. 190](#_Toc90641662)

[UML diagrama: 190](#_Toc90641663)

[Kodas: 191](#_Toc90641665)

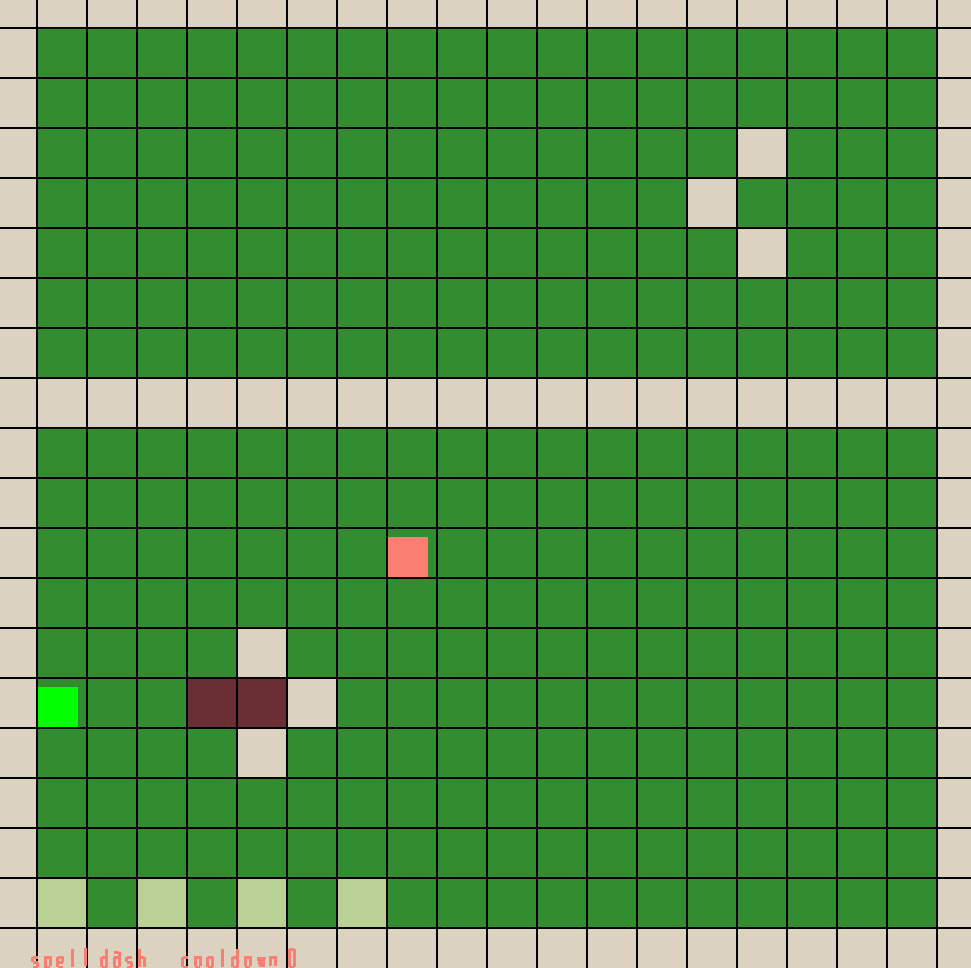
[Interpreter 204](#_Toc90641666)

[Chain of responsibility 216](#_Toc90641668)

[Composite 225](#_Toc90641669)

# PROJEKTO APRAŠYMAS

Žaidimas, kuriame strategiškai dėlioji bombas, kuriomis gali sunaikinti kliūtis bei priešus, išvengi- nėji spąstų bei naudoji įgytas galias kad įgautum pranašumą prieš savo varžovą.



### pav. Žaidimo prototipo nuotrauka

# ŽAIDIMO REIKALAVIMAI

# ŽAIDIMO LYGIAI

Žaidimas susidės iš trijų lygių, kurie vienas nuo kito skirsis savo žaidimo strategijomis bei sudėtin- gumu.

# PIRMASIS LYGIS

Jame bus sukurtas pasaulis, kuris susidės iš sienų, kurių kiaurai pereiti negalima, bet galima jas susprogdinti naudojant bombas, bei žemės, per kurią žaidėjas gales laisvai vaikščioti. Žaidimo tiklsas nugalėti savo priešininką, taktiškai naikinant sienas. Žaidėjai turės po 3 gyvybes, viena gyvybė yra pra- randama jeigu savo arba priešininko bomba sprogsta šalia.

# ANTRAS LYGIS

Antrame lygyje atsiranda nauja kliūtis- tai spąstai, kurie sugeneruojami sukuriant 2 lygio pasaulį. Visi spąstai atrodo taip pat tik žaidėjai nežino, ką jis gali padaryti, tik žino tą kad visi spąstai jį užsaldys. Taip pat spąstai gali žaidėja sulėtinti, nuimti gyvybę ar jį kažkur nukelti.

# TREČIAS LYGIS

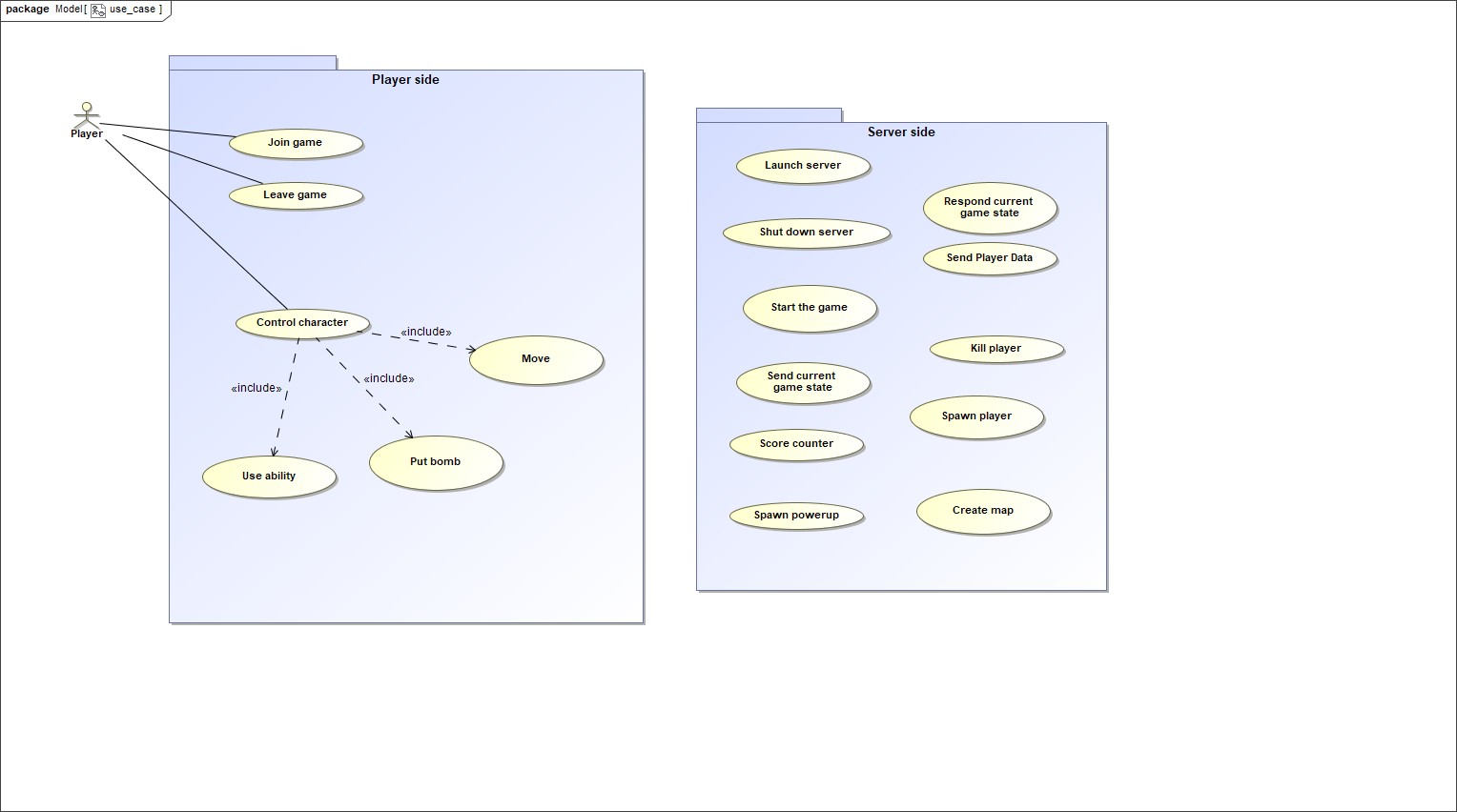
Trečiame lygyje atsiranda žaidėjo įgudis. Kai 3 lygio pasaulis yra sugeneruotas kas 10 sekundžių atsiranda žaidėjo įgudis, ant kurio užlipus žaidėjas gauna viena iš 4 pagerinimų:

* + Sulėtinimas priešininko. Galima naudoti kas 30 sekundžių.
  + Nusikelimas į naują vietą. Galima naudoti kas 20 sekundžių.
  + Sienų peršokimas. Galima naudoti kas 15 sekundžių.
  + Greitas žaidėjo paslinkimas. Galima naudoti kas 15 sekundžių.

# PROJEKTUI NAUDOTOS TECHNOLOGIJOS

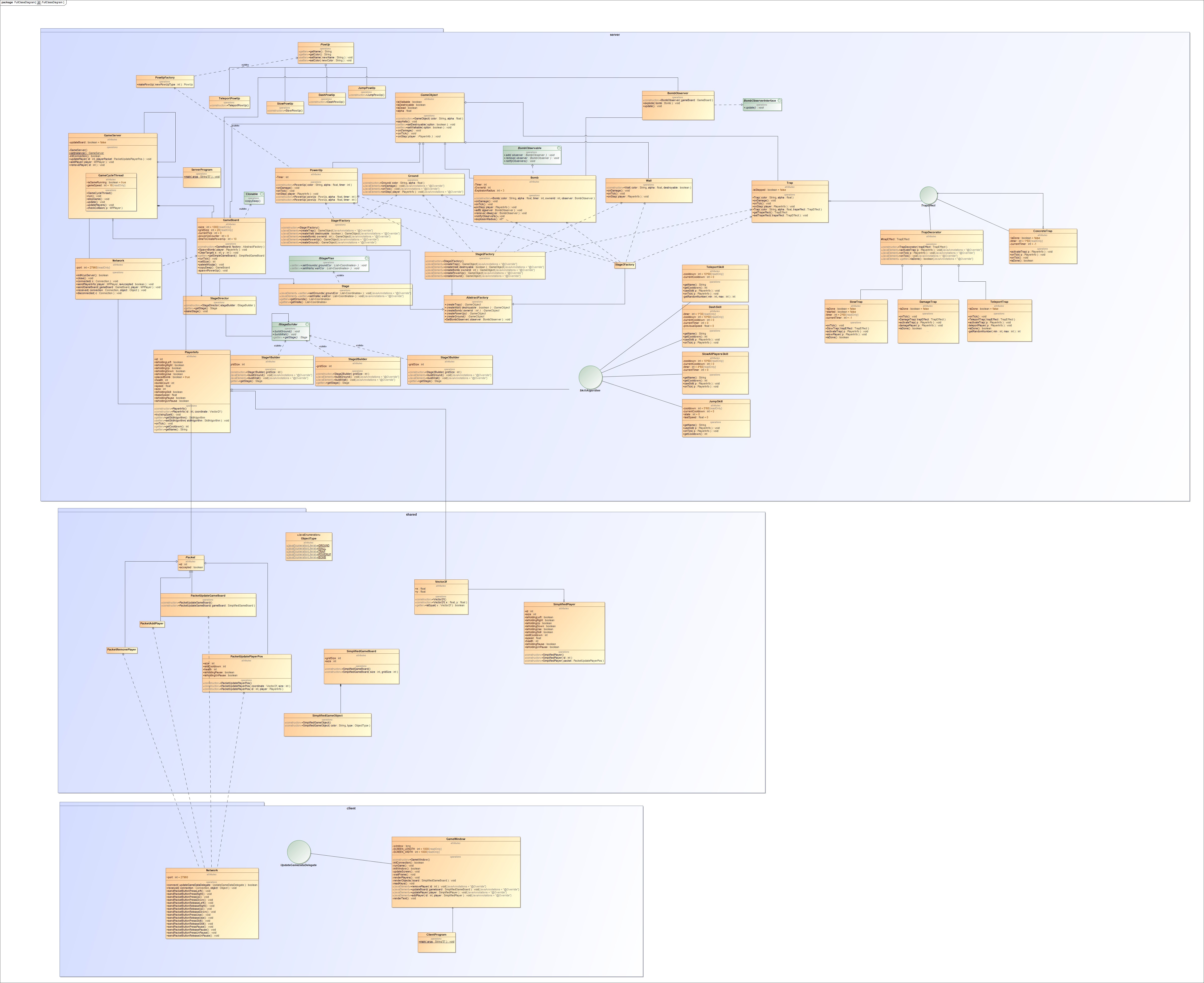
Serverio ir kliento komunikacijai naudotas Kryonet. Programos lango sukūrimui ir piešimui buvo pasitelkta lwjgl ir OpenGL 1.1 įrankiai.

# USE CASE DIAGRAMA



### 4.2 pav. Use case diagrama

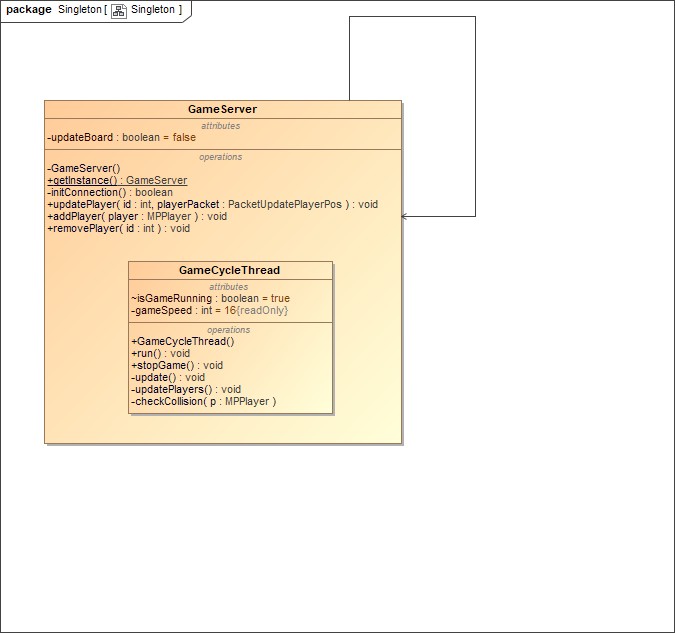
# KLASIŲ DIAGRAMA



### pav. Klasių diagrama

# 5.1. ŠABLONAI

## Singleton



### pav. Singleton diagrama

Šis algoritmas buvo pasirinktas dėl to, jog vienu metu gali vykti tik vienas žaidimas su bendra logika.

Taip pat šią žaidimo klasę paranku pasiekti visame projekte.

1 package s e r v e r ;

2

3 import java . u t i l . HashMap ;

4 import java . u t i l . Map;

5

6 import shared . Vector 2 f ;

7 import shared . PacketUpdate Player Pos ;

8

9 c l a s s GameServer

10 {

11 p r i v a te s t a t i c GameServer game Server = n u l l ;

12

13 p r o te c te d v o l a t i l e GameBoard gameBoard ;

14 p r o te c te d v o l a t i l e Map<I nteger , MPPlayer> p l a y e r s ;

15 p r o te c te d v o l a t i l e Network network ;

16 p r i v a te GameCycleThread thread ;

17 p r i v a te Stage 1 Factory s ta g e 1 f a c to r y ;

18 p r i v a te boolean updateBoard = f a l s e ;

19

20 p r i v a te GameServer ( )

21 {

22 // I n i t Connection

23 i f ( ! i n i t C o n n e c t i o n ( ) )

24 {

25 System . e r r . p r i n t l n ( "ERROR Connecting to host " ) ;

26 return ;

27 }

28

29 t h i s . p l a y e r s = new HashMap<Integer , MPPlayer >() ;

30 t h i s . s ta g e 1 f a c to r y = new Stage 1 Factory ( ) ;

31 t h i s . gameBoard = new GameBoard( s ta g e 1 f a c to r y ) ;

32

33

34

35 }

t h i s . thread = new GameCycleThread ( ) ; t h i s . thread . s t a r t ( ) ;

36

37 p u b l i c s t a t i c GameServer g e t I n s ta n c e ( )

38 {

39 i f ( game Server == n u l l )

40 {

41 game Server = new GameServer ( ) ;

42 }

43 return game Server ;

44 }

45

46 p r i v a te boolean i n i t C o n n e c t i o n ( )

47 {

48 t h i s . network = new Network ( ) ;

49

50 i f ( ! t h i s . network . i n i t Kr y o Se r v e r ( ) )

51 {

52 return f a l s e ;

53 }

54

55

56 }

return true ;

57

58 p u b l i c void update Player ( i n t id , PacketUpdate Player Pos player Packet )

59 {

60

61

62

63

64

65

66

67

68

69

70

71 }

MPPlayer p l a y e r = p l a y e r s . get ( id ) ; i f ( p l a y e r != n u l l )

{

p l a y e r . is Holding Up = player Packet . is Holding Up != n u l l ? player Packet . is Holding Up : p l a y e r . is Holding Up ;

p l a y e r . isHoldingDown = player Packet . isHoldingDown != n u l l ? player Packet . isHoldingDown : p l a y e r . isHoldingDown ;

p l a y e r . i s H o l d i n g L e f t = player Packet . i s H o l d i n g L e f t != n u l l ? player Packet . i s H o l d i n g L e f t : p l a y e r . i s H o l d i n g L e f t ;

p l a y e r . i s Holding Right = player Packet . i s Holding Right != n u l l ? player Packet . i s Holding Right : p l a y e r . i s Holding Right ;

p l a y e r . i s Holding Use = player Packet . i s Holding Use != n u l l ? player Packet . i s Holding Use : p l a y e r . i s Holding Use ;

p l a y e r . i s H o l d i n g S k i l l = player Packet . i s H o l d i n g S k i l l != n u l l ? player Packet . i s H o l d i n g S k i l l : p l a y e r . i s H o l d i n g S k i l l ;

p l a y e r s . put ( p l a y e r . id , p l a y e r ) ;

}

72

73 p u b l i c void add Player ( MPPlayer p l a y e r )

74 {

75 t h i s . p l a y e r s . put ( p l a y e r . c . getID ( ) , p l a y e r ) ;

76 t h i s . network . sendGameBoard ( gameBoard , p l a y e r ) ;

77

78 }

79

80 p u b l i c void remove Player ( i n t id )

81 {

82

83

84 }

p l a y e r s . remove ( id ) ;

85

86

87 p r i v a te c l a s s GameCycleThread extends Thread

88 {

89 v o l a t i l e boolean isGameRunning = true ;

90 p r i v a te f i n a l i n t gameSpeed = 1 6 ; //The lower the number the f a s t e r the game i s

91

92 p u b l i c GameCycleThread ( )

93 {

94 t h i s . isGameRunning = true ;

95 }

96

97 p u b l i c void run ( )

98 {

99 while ( t h i s . isGameRunning )

100 {

101 try

102 {

103 // Probably should use Timer i n s te a d

104 Thread . s l e e p ( gameSpeed ) ;

105 t h i s . update ( ) ;

106 }

107 catch ( Interrupted Exception e )

108 {

109 e . print Stack Trace ( ) ;

110 t h i s . stopGame ( ) ;

111 }

112 }

113 }

114

115 p u b l i c void stopGame ( )

116 {

117 t h i s . isGameRunning = f a l s e ;

118

119 // network should probably be c l o s e d by the parent

120 network . c l o s e ( ) ;

121 }

122

123 p r i v a te void update ( )

124 {

125 update Players ( ) ;

126 gameBoard . run Tick ( ) ;

127 }

128

129 p r i v a te void update Players ( )

130 {

131 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

132 {

133 i f ( p . i s Holding Pause )

134 {

135

136 }

137

138 i f ( p . i s H o l d i n g S k i l l )

139 {

140 p . tr y U s i n g S p e l l ( ) ;

141 }

142

143 p . onTick ( ) ;

144

145 i f ( p . i s Holding Use )

146 {

147 gameBoard . SpawnBomb( p ) ;

148 }

149

150 p . c o o r d i n a te = c h e c k C o l l i s i o n ( p ) ;

151

152 network . sendGameBoard ( gameBoard , p ) ;

153 network . se n d Player In f o ( p , true ) ;

154 }

155 }

156

157 p r i v a te Vector 2 f c h e c k C o l l i s i o n ( MPPlayer p )

158 {

159

160 Vector 2 f coords After Move = new Vector 2 f ( p . c o o r d i n a te . x , p . c o o r d i n a te . y ) ;

161

162 f l o a t padding = 0 . 001 f ;

163 f l o a t c e l l S i z e = gameBoard . c e l l S i z e ( ) ;

164

165 boolean moveX = true ;

166 boolean moveY = true ;

167

168 i f ( p . i s H o l d i n g L e f t )

169 {

170 coords After Move . x -= p . speed ;

171 }

172

173 i f ( p . i s Holding Right )

174 {

175

176 coords After Move . x += p . speed ;

177 }

178

179 boolean c o l l i d i n g L e f t = ( ( i n t ) coords After Move . x / c e l l S i z e - padding ) < ( ( i n t ) p . c o o r d i n a te . x / c e l l S i z e ) ;

180 boolean c o l l i d i n g R i g h t = ( ( ( i n t ) coords After Move . x + p . s i z e + padding ) / c e l l S i z e ) > ( ( ( i n t ) p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

181 boolean i s C o l l i d i n g X = c o l l i d i n g L e f t | | c o l l i d i n g R i g h t ;

182 moveX = ! ( coords After Move . x <= 0 | | coords After Move . x >= gameBoard . s i z e - p . s i z e ) ;

183 //Some smoothing when going around edges would be n i c e

184 i f ( i s C o l l i d i n g X && moveX)

185 {

186 i n t x = 0 , y = 0 , y1 = 0 ;

187 i f ( c o l l i d i n g L e f t )

188 {

189 x = ( i n t ) ( ( coords After Move . x ) / c e l l S i z e ) ;

190 y = ( i n t ) ( p . c o o r d i n a te . y / c e l l S i z e ) ;

191 y1 = ( i n t ) ( ( p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

192

193 }

194 i f ( c o l l i d i n g R i g h t )

195 {

196 x = ( i n t ) ( ( coords After Move . x + p . s i z e ) / c e l l S i z e ) ;

197 y = ( i n t ) ( p . c o o r d i n a te . y / c e l l S i z e ) ;

198 y1 = ( i n t ) ( ( p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

199

200 }

201

202 i f ( y == y1 )

203 {

204 moveX = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable ;

205 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

206 }

207 e l s e

208 {

209 moveX = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable && gameBoard . o b j e c t s [ x ] [ y1 ] . i s Walkable ;

210 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

211 gameBoard . o b j e c t s [ x ] [ y1 ] . onStep ( p ) ;

212 }

213 }

214

215

216 i f ( p . is Holding Up )

217 {

218 coords After Move . y += p . speed ;

219 }

220

221 i f ( p . isHoldingDown )

222 {

223 coords After Move . y -= p . speed ;

224 }

225

226 boolean c o l l i d i n g U p = ( ( ( i n t ) coords After Move . y + p . s i z e - padding

) / c e l l S i z e ) > ( ( ( i n t ) p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

227 boolean colliding Down = ( ( i n t ) coords After Move . y / c e l l S i z e + padding ) < ( ( i n t ) p . c o o r d i n a te . y / c e l l S i z e ) ;

228 boolean i s C o l l i d i n g Y = c o l l i d i n g U p | | colliding Down ;

229 moveY = ! ( coords After Move . y <= 0 | | coords After Move . y >= gameBoard . s i z e - p . s i z e ) ;

230

231 //Some smoothing when going around edges would be n i c e

232 i f ( i s C o l l i d i n g Y && moveY)

233 {

234 i n t x = 0 , x1 = 0 , y = 0 ;

235

236 i f ( c o l l i d i n g U p )

237 {

238 y = ( i n t ) ( ( coords After Move . y + p . s i z e ) / c e l l S i z e ) ;

239 x = ( i n t ) ( p . c o o r d i n a te . x / c e l l S i z e ) ;

240 x1 = ( i n t ) ( ( p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

241

242 }

243 i f ( colliding Down )

244 {

245 y = ( i n t ) ( ( coords After Move . y ) / c e l l S i z e ) ;

246 x = ( i n t ) ( p . c o o r d i n a te . x / c e l l S i z e ) ;

247 x1 = ( i n t ) ( ( p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

248 }

249

250 i f ( x == x1 )

251 {

252 moveY = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable ;

253 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

254 }

255 e l s e

256 {

257 moveY = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable && gameBoard . o b j e c t s [ x1 ] [ y ] . i s Walkable ;

258 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

259 gameBoard . o b j e c t s [ x1 ] [ y ] . onStep ( p ) ;

260 }

261 }

262

263 coords After Move . x = moveX ? coords After Move . x : p . c o o r d i n a te . x ;

264 coords After Move . y = moveY ? coords After Move . y : p . c o o r d i n a te . y ;

265

266 return coords After Move ;

267

268 }

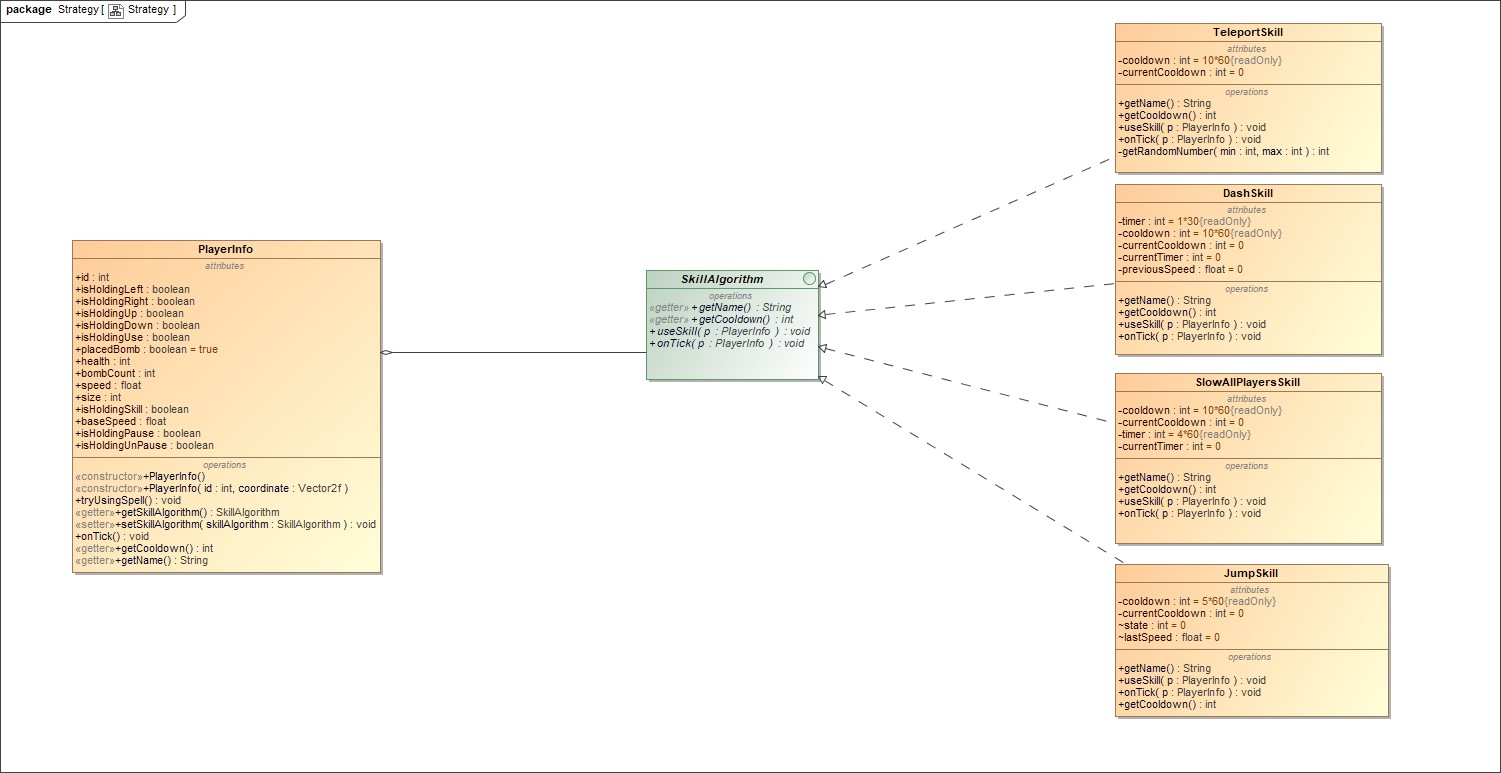
269

270 }

271

272 }

## Strategy



### pav. Strategy diagrama

Strategy šablono poreikis atsirado norint įterpti skirtinga funkcionalumą žaidėjui nepakeičiant kaip jis jį naudoja. Pasitelkti skirtingi algoritmai skirtingiems įgudžiams.

package s e r v e r ;

p u b l i c i n t e r f a c e S k i l l A l g o r i th m

{

i n t getCooldown ( ) ;

void u s e S k i l l ( Pl a y e r In f o p ) ; void onTick ( Pl a y e r In f o p ) ; S tr i n g getName ( ) ;

}

1

2

3

4

5

6

7

8

9

1

package s e r v e r ;

2

3 p u b l i c c l a s s Jump Skill implements S k i l l A l g o r i th m

4 {

5 p r i v a te f i n a l i n t cooldown = 5 \* 6 0 ; // 5 seconds

6 p r i v a te f i n a l S tr i n g name = " Jump" ;

7 p r i v a te i n t currentCooldown = 0 ;

8 i n t s ta te = 0 ;

9 f l o a t l a s t Sp e e d = 0 ;

10 @Override

11 p u b l i c void u s e S k i l l ( Pl a y e r In f o p )

12 {

13 i f ( t h i s . currentCooldown == 0 )

14 {

15 t h i s . s ta te = 2 ;

16 t h i s . currentCooldown = t h i s . cooldown ;

17 }

18 }

19

20 @Override

21 p u b l i c void onTick ( Pl a y e r In f o p )

22 {

23 i f ( t h i s . currentCooldown > 0 )

24 {

25 t h i s . currentCooldown - - ;

26 }

27

28 i f ( t h i s . s ta te == 2 )

29 {

30 GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ;

31 t h i s . l a s t Sp e e d = p . speed ;

32

33

34

35

36

37

38

39

40 }

p . speed = ( gameserver . gameBoard . s i z e / gameserver . gameBoard . g r i d S i z e ) \* 2 ;

t h i s . s ta te = 1 ;

}

e l s e i f ( t h i s . s ta te == 1 )

{

p . speed = l a s t Sp e e d ; t h i s . s ta te = 0 ;

}

41

42 @Override

43 p u b l i c S tr i n g getName ( )

44 {

45 return t h i s . name ;

46 }

47

48

49

50

51

52

53 }

@Override

p u b l i c i n t getCooldown ( )

{

return t h i s . currentCooldown ;

}

1

package s e r v e r ;

import java . u t i l . Array List ; import java . u t i l . HashMap ; import java . u t i l . L i s t ;

2

3

4

5

6 import java . u t i l . Map;

7

8 import shared . S i m p l i f i e d P l a y e r ;

9

10 p u b l i c c l a s s S l o w A l l P l a y e r s S k i l l implements S k i l l A l g o r i th m

11 {

12 p r i v a te f i n a l i n t cooldown = 10 \* 6 0 ; // 10 seconds

13 p r i v a te f i n a l S tr i n g name = " Slow " ;

14 p r i v a te i n t currentCooldown = 0 ;

15 p r i v a te f i n a l i n t timer = 4 \* 6 0 ; // 4 seconds

16 p r i v a te i n t current Timer = 0 ;

17 p r o te c te d v o l a t i l e Map<I nteger , S i m p l i f i e d Pl a y e r > s i m p l i f i e d P l a y e r s ;

18 @Override

19 p u b l i c void u s e S k i l l ( Pl a y e r In f o p )

20 {

21 i f ( t h i s . currentCooldown == 0 )

22 {

23 GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ;

24 s i m p l i f i e d P l a y e r s = new HashMap<Integer , S i m p l i f i e d Pl a y e r >() ;

25 f o r ( MPPlayer s i n g l e P l a y e r : gameserver . p l a y e r s . v a l u e s ( ) )

26 {

27 i f ( s i n g l e P l a y e r != n u l l )

28 {

29 i f ( s i n g l e P l a y e r != p )

30 {

31 S i m p l i f i e d P l a y e r p l a y e r = new S i m p l i f i e d P l a y e r ( ) ;

32 p l a y e r . id = p . id ;

33 p l a y e r . speed = p . speed ;

34 f l o a t newSpeed = p . speed \* 0 . 5 f ;

35

36

37 }

s i n g l e P l a y e r . speed = newSpeed ;

t h i s . s i m p l i f i e d P l a y e r s . put ( p l a y e r . id , p l a y e r ) ;

38

39 }

40

41

42

43

44 }

45 }

}

t h i s . current Timer = timer ;

t h i s . currentCooldown = t h i s . cooldown ;

46

47 @Override

48 p u b l i c void onTick ( Pl a y e r In f o p )

49 {

50 i f ( t h i s . currentCooldown > 0 )

51 {

52 t h i s . currentCooldown - - ;

53 }

54 i f ( t h i s . current Timer > 1 )

55 {

56 t h i s . currentTimer - - ;

57 }

58 e l s e i f ( t h i s . current Timer == 1 )

59 {

60 GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ;

61 f o r ( MPPlayer s i n g l e P l a y e r : gameserver . p l a y e r s . v a l u e s ( ) )

62 {

63 i f ( s i n g l e P l a y e r != n u l l )

64 {

65

66

67

i f ( s i n g l e P l a y e r != p )

{

S i m p l i f i e d P l a y e r s i m p l i f i e d P l a y e r = t h i s . s i m p l i f i e d P l a y e r s .

get ( s i n g l e P l a y e r . id ) ;

68 i f ( s i m p l i f i e d P l a y e r != n u l l )

69 {

70 s i n g l e P l a y e r . speed = s i m p l i f i e d P l a y e r . speed ;

71 }

72 }

73

74 }

75

76

77

78 }

}

t h i s . current Timer - - ;

79

80 }

81

82 @Override

83 p u b l i c S tr i n g getName ( )

84 {

85 return t h i s . name ;

86 }

87

88 @Override

89 p u b l i c i n t getCooldown ( )

90 {

91 return t h i s . currentCooldown ;

92

}

}

93

1 package s e r v e r ;

2

3 import shared . S i m p l i f i e d P l a y e r ;

4

5 p u b l i c c l a s s Das h Skill implements S k i l l A l g o r i th m

6 {

7

8 p r i v a te f i n a l i n t cooldown = 10 \* 6 0 ; // 10 seconds

9 p r i v a te f i n a l S tr i n g name = " Dash " ;

10 p r i v a te i n t currentCooldown = 0 ;

11 p r i v a te f i n a l i n t timer = 1 \* 3 0 ; // 0 . 25 second

12 p r i v a te i n t current Timer = 0 ;

13 p r i v a te S i m p l i f i e d P l a y e r s i m p l i f i e d ; // For d i r e c t i o n

14 p r i v a te f l o a t previous Speed = 0 ;

15 @Override

16 p u b l i c void u s e S k i l l ( Pl a y e r In f o p )

17 {

18 i f ( t h i s . currentCooldown == 0 )

19 {

20 t h i s . current Timer = timer ;

21 t h i s . currentCooldown = t h i s . cooldown ;

22 t h i s . previous Speed = p . speed ;

23 p . speed += 1 0 ;

24 t h i s . s i m p l i f i e d = new S i m p l i f i e d P l a y e r ( ) ;

25 t h i s . s i m p l i f i e d . isHoldingDown = p . isHoldingDown ;

26 t h i s . s i m p l i f i e d . is Holding Up = p . is Holding Up ;

27

28

29 }

30 }

t h i s . s i m p l i f i e d . i s H o l d i n g L e f t = p . i s H o l d i n g L e f t ; t h i s . s i m p l i f i e d . i s Holding Right = p . i s Holding Right ;

31

32 @Override

33 p u b l i c void onTick ( Pl a y e r In f o p )

34 {

35 i f ( t h i s . currentCooldown > 0 )

36 {

37 t h i s . currentCooldown - - ;

38 }

39

40 i f ( t h i s . current Timer > 1 )

41 {

42 p . isHoldingDown = t h i s . s i m p l i f i e d . isHoldingDown ;

43 p . is Holding Up = t h i s . s i m p l i f i e d . is Holding Up ;

44 p . i s H o l d i n g L e f t = t h i s . s i m p l i f i e d . i s H o l d i n g L e f t ;

45 p . i s Holding Right = t h i s . s i m p l i f i e d . i s Holding Right ;

46 t h i s . current Timer - - ;

47 }

48 e l s e i f ( t h i s . current Timer == 1 )

49 {

50 p . speed = t h i s . previous Speed ;

51 t h i s . current Timer - - ;

52 }

53

54 }

55

56

@Override

p u b l i c S tr i n g getName ( )

{

return t h i s . name ;

}

@Override

p u b l i c i n t getCooldown ( )

{

return t h i s . currentCooldown ;

}

}

57

58

59

60

61

62

63

64

65

66

67

1

package s e r v e r ;

p u b l i c c l a s s T e l e p o r t S k i l l implements S k i l l A l g o r i th m

{

p r i v a te f i n a l i n t cooldown = 10 \* 6 0 ; // 10 seconds p r i v a te f i n a l S tr i n g name = " Teleport " ;

p r i v a te i n t currentCooldown = 0 ;

@Override

p u b l i c void u s e S k i l l ( Pl a y e r In f o p )

{

i f ( t h i s . currentCooldown == 0 )

{

GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ;

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34 }

boolean te l e p o r te d = f a l s e ; i n t maxRetry = 6 0 ;

i n t r e t r y = 0 ;

while ( ! te l e p o r te d && r e t r y < maxRetry )

{

i n t randomCoordX = getRandomNumber ( 0 , gameserver . gameBoard . g r i d S i z e ) ;

i n t randomCoordY = getRandomNumber ( 0 , gameserver . gameBoard . g r i d S i z e ) ;

i f ( gameserver . gameBoard . o b j e c t s [ randomCoordX ] [ randomCoordY ] i n s t a n c e o f Ground )

{

p . c o o r d i n a te . x = randomCoordX \* ( gameserver . gameBoard . s i z e / gameserver . gameBoard . g r i d S i z e ) ;

p . c o o r d i n a te . y = randomCoordY \* ( gameserver . gameBoard . s i z e / gameserver . gameBoard . g r i d S i z e ) ;

te l e p o r te d = true ;

}

r e t r y ++;

}

t h i s . currentCooldown = t h i s . cooldown ;

}

35

36 @Override

37 p u b l i c void onTick ( Pl a y e r In f o p )

38 {

39 i f ( t h i s . currentCooldown > 0 )

40 {

41

42 }

43 }

t h i s . currentCooldown - - ;

44

45 @Override

46 p u b l i c S tr i n g getName ( )

47 {

48 return t h i s . name ;

49 }

50

51 @Override

52 p u b l i c i n t getCooldown ( )

53 {

54 return t h i s . currentCooldown ;

55 }

56

57 p r i v a te i n t getRandomNumber ( i n t min , i n t max)

58 {

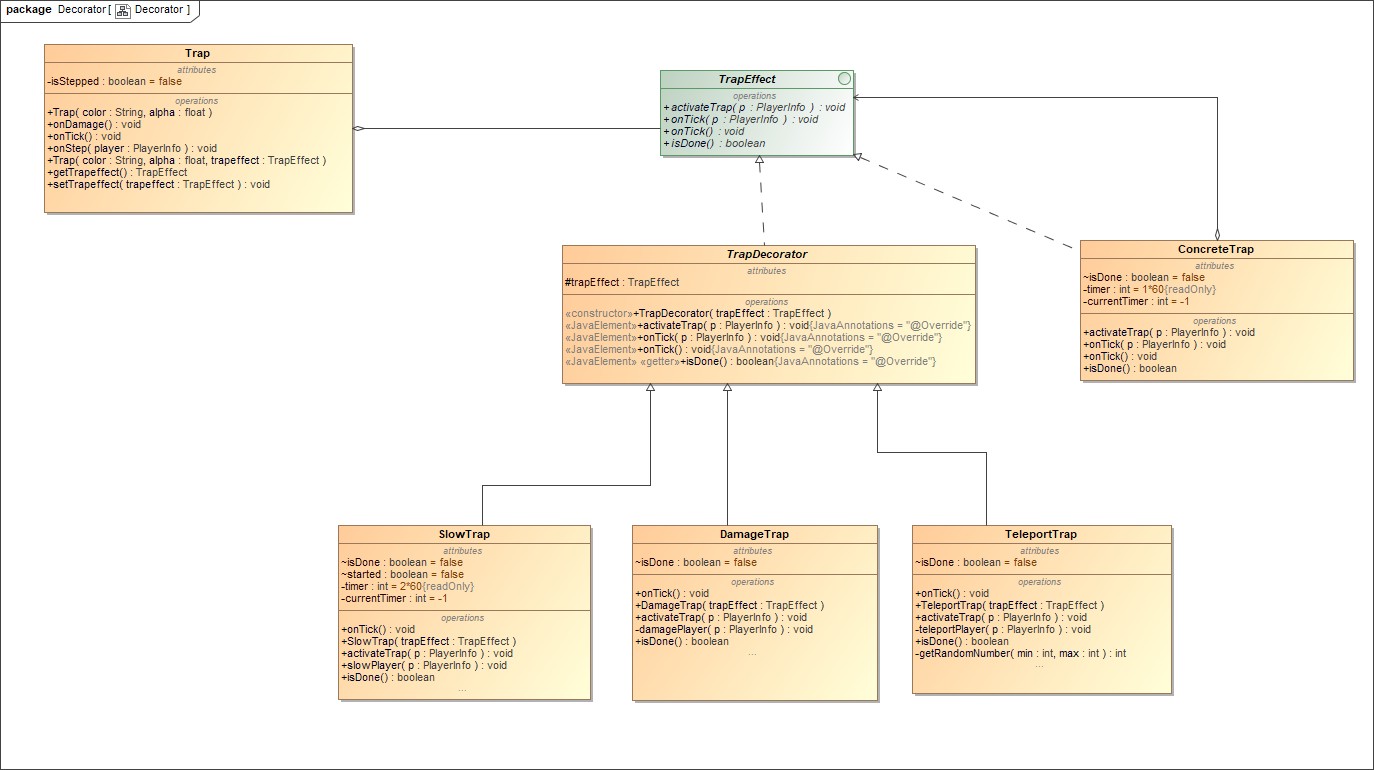
59 return ( i n t ) ( ( Math . random ( ) \* (max - min ) ) + min ) ;

60 }

61

62 }

## Decorator



### pav. Decorator diagrama

Decorator šablono poreikis atsirado norint turėti skirtingus spąstus, su skirtingais efektais, kurie atliekami žaidėjui. Šie efektai gali sluoksniuotis.

package s e r v e r ;

p u b l i c c l a s s Trap extends GameObject{

p r i v a te Trap Effect t r a p e f f e c t ; p r i v a te boolean i s Stepped = f a l s e ;

p u b l i c Trap ( S tr i n g c o l or , f l o a t alpha , Trap Effect t r a p e f f e c t )

{

1

2

3

4

5

6

7

8

9

10

11

12

13 }

super ( c o l or , alpha ) ;

t h i s . t r a p e f f e c t = t r a p e f f e c t ;

14

15 p u b l i c void onDamage ( )

16 {

17 i f ( t h i s . i s D e s tr o y a b l e )

18 is Dead = true ;

19 }

20 p u b l i c void onTick ( )

21 {

22 t h i s . t r a p e f f e c t . onTick ( ) ;

23 i f ( t h i s . t r a p e f f e c t . is Done ( ) )

24 {

25 is Dead = true ;

26 }

27 }

28

29 p u b l i c void onStep ( Pl a y e r In f o p l a y e r )

30 {

31 i f ( ! i s Stepped )

32 {

33 t h i s . t r a p e f f e c t . activate Trap ( p l a y e r ) ;

34 i s Stepped = true ;

35 }

36 }

37

38 p u b l i c Trap Effect g e t Tr a p e f f e c t ( )

39 {

40

return t r a p e f f e c t ;

}

p u b l i c void s e t T r a p e f f e c t ( Trap Effect t r a p e f f e c t )

{

t h i s . t r a p e f f e c t = t r a p e f f e c t ;

}

}

41

42

43

44

45

46

47

48

1

package s e r v e r ;

p u b l i c i n t e r f a c e Trap Effect

{

void activate Trap ( Pl a y e r In f o p ) ; void onTick ( ) ;

boolean is Done ( ) ;

}

2

3

4

5

6

7

8

1

package s e r v e r ;

p u b l i c c l a s s Concrete Trap implements Trap Effect

{

p r i v a te Pl a y e r In f o p ; boolean is Done = f a l s e ;

p r i v a te f i n a l i n t timer = 1 \* 6 0 ; // second p r i v a te i n t current Timer = - 1 ;

@Override

p u b l i c void activate Trap ( Pl a y e r In f o p )

{

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17 }

t h i s . p = p ;

t h i s . p . speed = 0 ;

GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ; gameserver . p l a y e r s . get ( t h i s . p . id ) . speed = t h i s . p . speed ; t h i s . current Timer = timer ;

18

19 @Override

20 p u b l i c void onTick ( )

21 {

22

23 i f ( t h i s . current Timer > 1 )

24 {

25 t h i s . current Timer - - ;

26 }

27 e l s e i f ( t h i s . current Timer == 1 )

28 {

29 GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ;

30 gameserver . p l a y e r s . get ( t h i s . p . id ) . speed = t h i s . p . base Speed ;

31 t h i s . current Timer - - ;

32 }

33 e l s e i f ( t h i s . current Timer == 0 )

34 {

35 t h i s . is Done = true ;

36 }

37

38 }

39

40 @Override

41

p u b l i c boolean is Done ( )

{

return is Done ;

}

}

42

43

44

45

46

1

package s e r v e r ;

p u b l i c a b s tr a c t c l a s s Trap Decorator implements Trap Effect

{

p r o te c te d Trap Effect t r a p E f f e c t ;

p u b l i c Trap Decorator ( Trap Effect t r a p E f f e c t )

{

t h i s . t r a p E f f e c t = t r a p E f f e c t ;

}

@Override

p u b l i c void activate Trap ( Pl a y e r In f o p )

{

t h i s . t r a p E f f e c t . activate Trap ( p ) ;

}

@Override

p u b l i c void onTick ( )

{

t h i s . t r a p E f f e c t . onTick ( ) ;

}

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

@Override

p u b l i c boolean is Done ( )

{

return t h i s . t r a p E f f e c t . is Done ( ) ;

}

}

24

25

26

27

28

29

30

1

package s e r v e r ;

p u b l i c c l a s s SlowTrap extends Trap Decorator

{

boolean is Done = f a l s e ; boolean s ta r te d = f a l s e ; p r i v a te Pl a y e r In f o p ;

p r i v a te f i n a l i n t timer = 2 \* 6 0 ; // 4 seconds p r i v a te i n t current Timer = - 1 ;

p u b l i c SlowTrap ( Trap Effect t r a p E f f e c t )

{

super ( t r a p E f f e c t ) ;

}

p u b l i c void activate Trap ( Pl a y e r In f o p )

{

super . activate Trap ( p ) ; t h i s . p = p ;

}

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

|  |  |  |  |
| --- | --- | --- | --- |
| 21 |  |  | |
| 22 | p u b l i c void onTick ( ) |
| 23 | { |
| 24 | super . onTick ( ) ; |
| 25 | s low Player ( p ) ; |
| 26 | } |
| 27 |  |
| 28 | p u b l i c void s low Player ( Pl a y e r In f o p ) |
| 29 | { |
| 30 | i f ( super . is Done ( ) && ! s ta r te d ) |
| 31 | { |
| 32 | t h i s . p . speed = 0 ; |
| 33 | GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ; |
| 34  35 | gameserver . p l a y e r s . get ( t h i s . p . id ) . speed = t h i s . p . base Speed  f ;  t h i s . current Timer = timer ; | \* | 0 . 5 |
| 36 | s ta r te d = true ; |  |  |
| 37 | } |  |  |
| 38 |  |  |  |
| 39 | i f ( t h i s . current Timer > 1 ) |  |  |
| 40 | { |  |  |
| 41 | t h i s . current Timer - - ; |  |  |
| 42 | } |  |  |
| 43 | e l s e i f ( t h i s . current Timer == 1 ) |  |  |
| 44 | { |  |  |
| 45 | GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ; |  |  |
| 46 | gameserver . p l a y e r s . get ( t h i s . p . id ) . speed = t h i s . p . base Speed ; |  |  |
| 47 | t h i s . current Timer - - ; |  |  |
| 48 | } |  |  |

49

e l s e i f ( t h i s . current Timer == 0 )

{

t h i s . is Done = true ;

}

}

p u b l i c boolean is Done ( )

{

return super . is Done ( ) && is Done ;

}

}

50

51

52

53

54

55

56

57

58

1

package s e r v e r ;

p u b l i c c l a s s DamageTrap extends Trap Decorator

{

boolean is Done = f a l s e ;

p u b l i c DamageTrap ( Trap Effect t r a p E f f e c t )

{

super ( t r a p E f f e c t ) ;

}

p u b l i c void activate Trap ( Pl a y e r In f o p )

{

super . activate Trap ( p ) ; damagePlayer ( p ) ;

}

p r i v a te void damagePlayer ( Pl a y e r In f o p )

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

|  |  |  |
| --- | --- | --- |
| 19  20  21  22  23  24 | {  GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ; gameserver . p l a y e r s . get ( p . id ) . health - - ;  is Done = true ;  } | |
| 25 | p u b l i c void onTick ( ) |  |
| 26 | { |  |
| 27 | super . onTick ( ) ; |  |
| 28 | } |  |
| 29 |  |  |
| 30 | @Override |  |
| 31 | p u b l i c boolean is Done ( ) |  |
| 32 | { |  |
| 33 | return super . is Done ( ) | && is Done ; |
| 34 | } |  |
| 35 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c c l a s s TeleportTrap | extends Trap Decorator |
| 4 | { |  |
| 5 |  |  |
| 6  7  8  9  10  11 | boolean is Done = f a l s e ;  p u b l i c TeleportTrap ( Trap Effect t r a p E f f e c t )  {  super ( t r a p E f f e c t ) ;  } | |

12 p u b l i c void activate Trap ( Pl a y e r In f o p )

13 {

14 super . activate Trap ( p ) ;

15 te l e p o r t P l a y e r ( p ) ;

16

17 }

18

19 p r i v a te void te l e p o r t P l a y e r ( Pl a y e r In f o p )

20 {

21 GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ;

22 boolean te l e p o r te d = f a l s e ;

23 i n t maxRetry = 6 0 ;

24 i n t r e t r y = 0 ;

25 while ( ! te l e p o r te d && r e t r y < maxRetry )

26 {

27 i n t randomCoordX = getRandomNumber ( 0 , gameserver . gameBoard . g r i d S i z e ) ;

28 i n t randomCoordY = getRandomNumber ( 0 , gameserver . gameBoard . g r i d S i z e ) ;

29 i f ( gameserver . gameBoard . o b j e c t s [ randomCoordX ] [ randomCoordY ] i n s t a n c e o f Ground )

30 {

31 p . c o o r d i n a te . x = randomCoordX \* ( gameserver . gameBoard . s i z e / gameserver . gameBoard . g r i d S i z e ) ;

32 p . c o o r d i n a te . y = randomCoordY \* ( gameserver . gameBoard . s i z e / gameserver . gameBoard . g r i d S i z e ) ;

33 te l e p o r te d = true ;

34 }

35 r e t r y ++;

36

}

is Done = true ;

}

p u b l i c void onTick ( )

{

super . onTick ( ) ;

}

@Override

p u b l i c boolean is Done ( )

{

return super . is Done ( ) && is Done ;

}

p r i v a te i n t getRandomNumber ( i n t min , i n t max)

{

return ( i n t ) ( ( Math . random ( ) \* (max - min ) ) + min ) ;

}

}

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

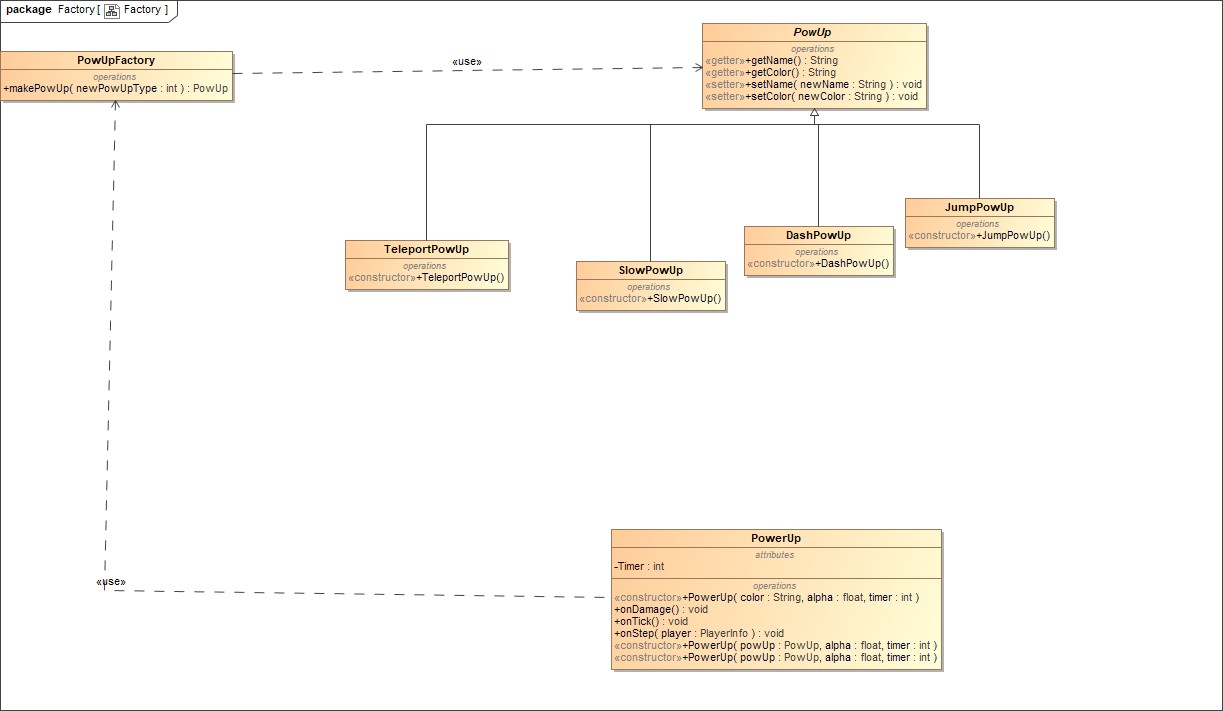
52

53

54

55

## Abstract Factory



### pav. Abstract Factory diagrama

Abstract Factory šablonas padeda lansksčiai sukurti skirtingus žaidimo objektų rinkinius skirtin- giems lygiams.

package s e r v e r ;

p u b l i c a b s tr a c t c l a s s Abstract Factory

{

p r o te c te d BombObserver bombObserver ; p u b l i c a b s tr a c t GameObject create Trap ( ) ;

p u b l i c a b s tr a c t GameObject create Wall ( boolean d e s tr o y a b l e ) ; p u b l i c a b s tr a c t GameObject createBomb ( i n t ownerid ) ;

p u b l i c a b s tr a c t GameObject createPowerUp ( ) ;

p u b l i c a b s tr a c t GameObject create Ground ( ) ;

1

2

3

4

5

6

7

8

9

10

11

p u b l i c void SetBombObserver ( BombObserver o b s e rv e r )

{

t h i s . bombObserver = o b s e r v e r ;

}

}

12

13

14

15

16

1

package s e r v e r ;

import java . u t i l . Hashtable ; import java . u t i l . Random ;

p u b l i c c l a s s Stage 1 Factory extends Abstract Factory { p r i v a te Hashtable<String , String > c o l o r s ;

p u b l i c Stage 1 Factory ( ) {

t h i s . c o l o r s = new Hashtable <>() ; t h i s . c o l o r s . put ( " Wall " , "#2e 2203 " ) ;

t h i s . c o l o r s . put ( " DesWall " , "#8c 8674 " ) ; t h i s . c o l o r s . put ( " Ground " , " #348C31 " ) ; t h i s . c o l o r s . put ( "Bomb" , "#6A2E35 " ) ; t h i s . c o l o r s . put ( " PowerUp " , "#BAD094 " ) ; t h i s . c o l o r s . put ( " Trap " , " #373D20 " ) ;

}

@Override

p u b l i c GameObject create Trap ( )

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

|  |  |  |
| --- | --- | --- |
| 23 |  | { |
| 24  25 | ; | return new Trap ( t h i s . c o l o r s . get ( " Trap " ) , 1 , new Concrete Trap ( ) )  } |
| 26 |  |  |
| 27 |  | @Override |
| 28 |  | p u b l i c GameObject create Wall ( boolean d e s tr o y a b l e ) |
| 29 |  | { |
| 30 |  | S tr i n g wallType = " Wall " ; |
| 31 |  |  |
| 32 |  | i f ( d e s tr o y a b l e ) |
| 33 |  | wallType = " DesWall " ; |
| 34 |  |  |
| 35 |  | return new Wall ( t h i s . c o l o r s . get ( wallType ) , 1 , d e s tr o y a b l e ) ; |
| 36 |  | } |
| 37 |  |  |
| 38 |  | @Override |
| 39 |  | p u b l i c GameObject createBomb ( i n t ownerid ) { |
| 40  41 | b | return new Bomb( c o l o r s . get ( "Bomb" ) , 1 , 90 , ownerid , t h i s . ombObserver ) ;  } |
| 42 |  |  |
| 43 |  | @Override |
| 44 |  | p u b l i c GameObject createPowerUp ( ) { |
| 45 |  |  |
| 46 |  | PowUpFactory powFactory = new PowUpFactory ( ) ; |
| 47 |  | PowUp powUp = n u l l ; |
| 48 |  |  |
| 49 |  | Random rand = new Random ( ) ; |

|  |  |  |  |
| --- | --- | --- | --- |
| 50 | i n t randomNum = rand . next Int (( 4 - 1 ) + 1 ) | + | 1 ; |
| 51 |  |  |  |
| 52 | powUp = powFactory . makePowUp(randomNum) ; |  |  |
| 53 |  |  |  |
| 54 |  |  |  |
| 55 | return new PowerUp(powUp , 1 , 5 ) ; |  |  |
| 56 | } |  |  |
| 57 |  |  |  |
| 58 | @Override |  |  |
| 59 | p u b l i c GameObject create Ground ( ) |  |  |
| 60 | { |  |  |
| 61 | return new Ground ( c o l o r s . get ( " Ground " ) , 1 ) ; |  |  |
| 62 | } |  |  |
| 63 |  |  |  |
| 64 | } |  |  |
|  |  |  |  |
| 1 | package s e r v e r ; |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4  5  6  7  8  9  10  11  12  13 | import java . u t i l . Hashtable ; import java . u t i l . Random ;  p u b l i c c l a s s Stage 2 Factory extends Abstract Factory { p r i v a te Hashtable<String , String > c o l o r s ;  p u b l i c Stage 2 Factory ( ) {  t h i s . c o l o r s = new Hashtable <>() ; t h i s . c o l o r s . put ( " Wall " , "#ECE5F0 " ) ; | | |

|  |  |  |
| --- | --- | --- |
| 14 |  | t h i s . c o l o r s . put ( " DesWall " , "#CEB5A7" ) ; |
| 15 |  | t h i s . c o l o r s . put ( " Ground " , "#FFA3AF" ) ; |
| 16 |  | t h i s . c o l o r s . put ( "Bomb" , " #210124 " ) ; |
| 17 |  | t h i s . c o l o r s . put ( " PowerUp " , "#87FF65 " ) ; |
| 18 |  | t h i s . c o l o r s . put ( " Trap " , "#8EA604 " ) ; |
| 19 |  | } |
| 20 |  |  |
| 21 |  | @Override |
| 22 |  | p u b l i c GameObject create Trap ( ) |
| 23 |  | { |
| 24  25 | ; | return new Trap ( t h i s . c o l o r s . get ( " Trap " ) , 1 , new Concrete Trap ( ) )  } |
| 26 |  |  |
| 27 |  | @Override |
| 28 |  | p u b l i c GameObject create Wall ( boolean d e s tr o y a b l e ) |
| 29 |  | { |
| 30 |  | S tr i n g wallType = " Wall " ; |
| 31 |  |  |
| 32 |  | i f ( d e s tr o y a b l e ) |
| 33 |  | wallType = " DesWall " ; |
| 34 |  |  |
| 35 |  | return new Wall ( t h i s . c o l o r s . get ( wallType ) , 1 , d e s tr o y a b l e ) ; |
| 36 |  | } |
| 37 |  |  |
| 38 |  | @Override |
| 39  40 | p u b l i c GameObject createBomb ( i n t ownerid ) {  return new Bomb( c o l o r s . get ( "Bomb" ) , 1 , 90 , ownerid , t h i s . bombObserver ) ; | |

41 }

42

43 @Override

44 p u b l i c GameObject createPowerUp ( ) {

45

46 PowUpFactory powFactory = new PowUpFactory ( ) ;

47 PowUp powUp = n u l l ;

48

49 Random rand = new Random ( ) ;

50 i n t randomNum = rand . next Int (( 4 - 1 ) + 1 ) + 1 ;

51

52 powUp = powFactory . makePowUp(randomNum) ;

53

54

55

56 }

return new PowerUp(powUp , 1 , 5 ) ;

57

58 @Override

59 p u b l i c GameObject create Ground ( )

60 {

61 return new Ground ( c o l o r s . get ( " Ground " ) , 1 ) ;

62 }

63

64 }

1

package s e r v e r ;

p u b l i c a b s tr a c t c l a s s Trap Decorator implements Trap Effect

{

2

3

4

5 p r o te c te d Trap Effect t r a p E f f e c t ;

6

7 p u b l i c Trap Decorator ( Trap Effect t r a p E f f e c t )

8 {

9 t h i s . t r a p E f f e c t = t r a p E f f e c t ;

10 }

11

12 @Override

13 p u b l i c void activate Trap ( Pl a y e r In f o p )

14 {

15 t h i s . t r a p E f f e c t . activate Trap ( p ) ;

16 }

17

18 @Override

19 p u b l i c void onTick ( )

20 {

21 t h i s . t r a p E f f e c t . onTick ( ) ;

22 }

23

24 @Override

25 p u b l i c boolean is Done ( )

26 {

27 return t h i s . t r a p E f f e c t . is Done ( ) ;

28 }

29

30 }

1

package s e r v e r ;

import java . u t i l . Random ;

2

3

4 p u b l i c c l a s s PowerUp extends GameObject

5 {

6 p r i v a te i n t Timer ;

7 p r i v a te PowUp powUp ;

8

9 p u b l i c PowerUp(PowUp powUp , f l o a t alpha , i n t timer ) {

10

11

12 super (powUp . get Color ( ) , alpha ) ;

13 t h i s . powUp = powUp ;

14 t h i s . Timer = timer ;

15 t h i s . i s Walkable = true ;

16

17

18

19 }

20

21 p u b l i c void onDamage ( ) {

22 i f ( t h i s . i s D e s tr o y a b l e )

23 System . out . p r i n t l n ( " Wall has been destroyed " ) ;

24 }

25 p u b l i c void onTick ( ) {

26 // t h i s . Timer - - ;

27 // i f ( t h i s . Timer <= 0 )

28 // is Dead = true ;

29 }

30

31

32

33

34

35

36

37

38 p u b l i c void onStep ( Pl a y e r In f o p l a y e r ) {

39 i f ( t h i s . i s Walkable )

40 {

41 i f ( ! is Dead )

42 {

43 GameServer game Server = GameServer . g e t I n s ta n c e ( ) ;

44

45

46 i f (powUp . getName ( ) . e q u a l s ( " Jump" ) )

47 game Server . p l a y e r s . get ( p l a y e r . id ) . s e t S k i l l A l g o r i th m ( new Jump Skill ( ) ) ;

48

49 e l s e i f (powUp . getName ( ) . e q u a l s ( " Dash " ) )

50 game Server . p l a y e r s . get ( p l a y e r . id ) . s e t S k i l l A l g o r i th m ( new Dash Skill ( ) ) ;

51

52 e l s e i f (powUp . getName ( ) . e q u a l s ( " Slow " ) )

53 game Server . p l a y e r s . get ( p l a y e r . id ) . s e t S k i l l A l g o r i th m ( new S l o w A l l P l a y e r s S k i l l ( ) ) ;

54

55 e l s e i f (powUp . getName ( ) . e q u a l s ( " Teleport " ) )

56 game Server . p l a y e r s . get ( p l a y e r . id ) . s e t S k i l l A l g o r i th m ( new T e l e p o r t S k i l l ( ) ) ;

57

is Dead = true ;

}

}

System . out . p r i n t l n ( " You gain powerup ! " ) ;

}

}

58

59

60

61

62

63

64

65

66

1

package s e r v e r ;

p u b l i c c l a s s Ground extends GameObject{

p u b l i c Ground ( S tr i n g c o l or , super ( c o l or , alpha ) ; t h i s . i s Walkable = true ;

}

f l o a t

alpha ) {

@Override

p u b l i c void onDamage ( ) {

}

@Override

p u b l i c void onTick ( ) {

}

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

@Override

p u b l i c void onStep ( Pl a y e r In f o p l a y e r ) {

}

}

20

21

22

23

24

1 package s e r v e r ;

2

3 import java . u t i l . L i s t ;

4 import java . u t i l . Stack ;

5

6 p u b l i c c l a s s Bomb extends GameObject implements BombObservable{

7

8 p r i v a te i n t Timer ;

9 p r i v a te i n t OwnerId ;

10 p r i v a te i n t Explosion Radius = 3 ;

11 p r i v a te List <BombObserver> o b s e r v e r s = new Stack<BombObserver >() ;

12

13 p u b l i c Bomb( S tr i n g color , f l o a t alpha , i n t timer , i n t ownerid , BombObserver o b s e r v e r ) {

14 super ( c o l or , alpha ) ;

15 t h i s . Timer = timer ;

16 t h i s . OwnerId = ownerid ;

17 t h i s . add ( o b s e r v e r ) ;

18 t h i s . i s Walkable = true ;

19 }

20

21 p u b l i c void onDamage ( ) {

22

23 }

// cannot be destroyed ?

24 p u b l i c void onTick ( ) {

25 t h i s . Timer - - ;

26 i f ( t h i s . Timer <= 0 )

27 n o t i f y O b s e r v e r s ( ) ;

28 }

29

30 p u b l i c void onStep ( Pl a y e r In f o p l a y e r ) {

31 // cannot be stepped on?

32 }

33

34 p u b l i c void add ( BombObserver o b s e rv e r )

35 {

36 t h i s . o b s e r v e r s . add ( ob s e r v e r ) ;

37 }

38

39 p u b l i c void remove ( BombObserver o b s e r v e r )

40 {

41 t h i s . o b s e r v e r s . remove ( o b s e r ve r ) ;

42 }

43

44 p u b l i c void n o t i f y O b s e r v e r s ( )

45 {

46 f o r ( BombObserver o b s e r v e r :

47 o b s e r v e r s ) {

48 o b s e r ve r . explode ( t h i s ) ;

49

50 }

51

}

p u b l i c i n t explosion Radius ( ) {

i f ( t h i s . Timer <= 0 ) return t h i s . Explosion Radius ; e l s e return 0 ;

}

}

52

53

54

55

56

57

1

package s e r v e r ;

p u b l i c c l a s s Wall extends GameObject{

p u b l i c Wall ( S tr i n g c o l or , f l o a t alpha , super ( c o l or , alpha ) ;

t h i s . i s D e s tr o y a b l e = d e s tr o y a b l e ;

}

boolean

d e s tr o y a b l e ) {

p u b l i c void onDamage ( )

{

i f ( t h i s . i s D e s tr o y a b l e ) is Dead = true ;

}

p u b l i c void onTick ( )

{

}

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

p u b l i c void onStep ( Pl a y e r In f o p l a y e r )

{

i f ( ! t h i s . i s Walkable )

System . out . p r i n t l n ( " Can ’ t walk on t h i s ! " ) ;

}

}

23

24

25

26

27

1

package s e r v e r ;

p u b l i c c l a s s Trap extends GameObject{

p r i v a te Trap Effect t r a p e f f e c t ;

p r i v a te boolean i s Stepped = f a l s e ;

p u b l i c Trap ( S tr i n g c o l or , f l o a t alpha ,

{

super ( c o l or , alpha ) ;

t h i s . t r a p e f f e c t = t r a p e f f e c t ;

}

Trap Effect

t r a p e f f e c t )

p u b l i c void onDamage ( )

{

i f ( t h i s . i s D e s tr o y a b l e ) is Dead = true ;

}

p u b l i c void onTick ( )

{

t h i s . t r a p e f f e c t . onTick ( ) ;

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27 }

i f ( t h i s . t r a p e f f e c t . is Done ( ) )

{

is Dead = true ;

}

28

29 p u b l i c void onStep ( Pl a y e r In f o p l a y e r )

30 {

31 i f ( ! i s Stepped )

32 {

33 t h i s . t r a p e f f e c t . activate Trap ( p l a y e r ) ;

34 i s Stepped = true ;

35 }

36 }

37

38

39

40

41 }

p u b l i c Trap Effect g e t Tr a p e f f e c t ( )

{

return t r a p e f f e c t ;

42

43 p u b l i c void s e t T r a p e f f e c t ( Trap Effect t r a p e f f e c t )

44 {

45 t h i s . t r a p e f f e c t = t r a p e f f e c t ;

46 }

47

48 }

1

package s e r v e r ;

2

3 p u b l i c a b s tr a c t c l a s s GameObject {

4 p u b l i c boolean i s Walkable ;

5 p u b l i c boolean i s D e s tr o y a b l e ;

6 p u b l i c boolean is Dead ;

7

8 GameObjectDelegate gameobjectdelegate ;

9

10 p u b l i c S tr i n g c o l o r ;

11 p u b l i c f l o a t alpha ;

12

13 p u b l i c GameObject ( S tr i n g color , f l o a t alpha ) {

14 super ( ) ;

15 t h i s . alpha = alpha ;

16 t h i s . c o l o r = c o l o r ;

17 }

18

19 p u b l i c void say Hello ( ) {

20 System . out . p r i n t l n ( " GameObject " ) ;

21 }

22

23 p u b l i c void s e t D e s tr o y a b l e ( boolean option ) {

24 t h i s . i s D e s tr o y a b l e = option ;

25 }

26

27 p u b l i c void set Walkable ( boolean option ) {

28 t h i s . i s Walkable = option ;

29 }

30

31 p u b l i c a b s tr a c t void onDamage ( ) ;

32

p u b l i c a b s tr a c t void onTick ( ) ;

p u b l i c a b s tr a c t void onStep ( Pl a y e r In f o p l a y e r ) ;

}

33

34

35

1 package s e r v e r ;

2

3 import java . u t i l . Array List ;

4 import java . u t i l . Random ;

5

6 import shared . \* ;

7

8 p u b l i c c l a s s GameBoard implements Cloneable {

9

10 p u b l i c f i n a l i n t s i z e = 1000 ;

11 p u b l i c f i n a l i n t g r i d S i z e = 2 0 ;

12 p u b l i c GameObject [ ] [ ] o b j e c t s ;

13 p r i v a te BombObserver bombObserver ;

14 p r i v a te Abstract Factory f a c to r y ;

15

16 p r i v a te i n t current Tick = 0 ;

17 p r i v a te i n t powerUpCounter = 0 ;

18 p r i v a te i n t timeToCreatePowerUp = 1 0 ;

19

20 p r i v a te ISta g e Bu i l d e r s ta g e 1 b u i l d e r ;

21 p r i v a te ISta g e Bu i l d e r s ta g e 2 b u i l d e r ;

22 p r i v a te ISta g e Bu i l d e r s ta g e 3 b u i l d e r ;

23

24 p u b l i c GameBoard( Abstract Factory f a c to r y )

25 {

26

27

28

29

30

t h i s . f a c to r y = f a c to r y ;

t h i s . o b j e c t s = new GameObject [ g r i d S i z e ] [ g r i d S i z e ] ; t h i s . s ta g e 1 b u i l d e r = new Stage 1 Builder ( g r i d S i z e ) ; t h i s . s ta g e 2 b u i l d e r = new Stage 2 Builder ( g r i d S i z e ) ; t h i s . s ta g e 3 b u i l d e r = new Stage 3 Builder ( g r i d S i z e ) ;

31

32 Sta g e D i r e c to r s ta g e D i r e c to r = new Sta g e D i r e c to r ( s ta g e 3 b u i l d e r ) ;

33

34 s ta g e D i r e c to r . makeStage ( ) ;

35

36 Stage s tage 1 = s ta g e D i r e c to r . get Stage ( ) ;

37

38 f o r ( Coordinates kor : s tage 1 . getGrounds ( ) )

39 t h i s . o b j e c t s [ kor . getX ( ) ] [ kor . getY ( ) ] = t h i s . f a c to r y . create Ground ( ) ;

40

41 i n t k i e k i s = 7 6 ;

42 boolean sunaikinama = f a l s e ;

43

44 f o r ( Coordinates kor : s tage 1 . get Walls ( ) ) {

45

46 t h i s . o b j e c t s [ kor . getX ( ) ] [ kor . getY ( ) ] = t h i s . f a c to r y . create Wall ( sunaikinama ) ;

47 k i e k i s - =1 ;

48 i f ( k i e k i s == 0 ) sunaikinama = true ;

49 }

50

51

52

53 bombObserver = new BombObserver ( t h i s ) ;

54

55

56 t h i s . f a c to r y . SetBombObserver ( bombObserver ) ;

57

58

59

60

61

62 // t e s t

63

64 //TODO wrong but ok f o r now

65 t h i s . o b j e c t s [ 1 7 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

66

67 t h i s . o b j e c t s [ 1 5 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

68 Trap modified = ( Trap ) t h i s . o b j e c t s [ 1 5 ] [ 3 ] ;

69 modified . s e t T r a p e f f e c t ( new DamageTrap ( new Concrete Trap ( ) ) ) ;

70 t h i s . o b j e c t s [ 1 5 ] [ 3 ] = modified ;

71

72 t h i s . o b j e c t s [ 1 3 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

73 Trap modified 1 = ( Trap ) t h i s . o b j e c t s [ 1 3 ] [ 3 ] ;

74 modified 1 . s e t T r a p e f f e c t ( new TeleportTrap ( new Concrete Trap ( ) ) ) ;

75 t h i s . o b j e c t s [ 1 3 ] [ 3 ] = modified 1 ;

76

77 t h i s . o b j e c t s [ 1 1 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

78 Trap modified 2 = ( Trap ) t h i s . o b j e c t s [ 1 1 ] [ 3 ] ;

79 modified 2 . s e t T r a p e f f e c t ( new SlowTrap ( new Concrete Trap ( ) ) ) ;

80 t h i s . o b j e c t s [ 1 1 ] [ 3 ] = modified 2 ;

81

82 t h i s . o b j e c t s [ 9 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

83 Trap modified 3 = ( Trap ) t h i s . o b j e c t s [ 9 ] [ 3 ] ;

84 modified 3 . s e t T r a p e f f e c t ( new DamageTrap ( new TeleportTrap ( new SlowTrap ( new Concrete Trap ( ) ) ) ) ) ;

85 t h i s . o b j e c t s [ 9 ] [ 3 ] = modified 3 ;

86

87 }

88

89 p u b l i c void p a l e i s t i K o p i j a ( ) {

90 GameBoard copy = copyDeep ( ) ;

91

92

93 }

94

95

96

97 p u b l i c GameBoard copyDeep ( ) {

98

99 try {

100 return ( GameBoard) t h i s . c l o n e ( ) ;

101 } catch ( Clone NotSupported Exception e ) {

102 e . print Stack Trace ( ) ;

103 return n u l l ;

104 }

105 }

106

107 p u b l i c void SpawnBomb( Pl a y e r In f o p l a y e r )

108 {

109 i n t x = Math . round ( p l a y e r . c o o r d i n a te . x /( s i z e / g r i d S i z e ) ) ;

110 i n t y = Math . round ( p l a y e r . c o o r d i n a te . y /( s i z e / g r i d S i z e ) ) ;

111 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . createBomb ( p l a y e r . id ) ;

112

113 }

114

115 p u b l i c void Clear Target ( i n t x , i n t y ) {

116 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . create Ground ( ) ;

117 System . out . p r i n t l n ( " Removing bomb from l o c a t i o n " + x + " " + y

) ;

118 }

119

120 p u b l i c Simplified Game Board getSimpleGameBoard ( )

121 {

122 Simplified Game Board simpleGameboard = new Simplified Game Board ( t h i s . s i z e , t h i s . g r i d S i z e ) ;

123

124 f o r ( i n t i = 0 ; i < t h i s . g r i d S i z e ; i ++)

125 {

126 f o r ( i n t j = 0 ; j < t h i s . g r i d S i z e ; j ++)

127 {

128 simpleGameboard . o b j e c t s [ i ] [ j ] = new Simplified Game Object ( t h i s . o b j e c t s [ i ] [ j ] . col or , ObjectType .GROUND) ;

129 }

130 }

131 return simpleGameboard ;

132 }

133

134 p r i v a te void spawnPowerUp ( ) {

135 Random rand = new Random ( ) ;

136

137 i f ( powerUpCounter > 4 )

138 return ;

139

140 while ( true ) {

141 i n t x = rand . next Int (( 19 - 0 ) + 1 ) + 0 ;

142 i n t y = rand . next Int (( 19 - 0 ) + 1 ) + 0 ;

143

144

145 i f ( t h i s . o b j e c t s [ x ] [ y ] i n s t a n c e o f Ground ) {

146 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . createPowerUp ( ) ;

147 powerUpCounter+=1;

148 break ;

149 }

150

151 }

152

153

154

155

156

157

158

159

160 }

161

162

163 p u b l i c void run Tick ( )

164 {

165 current Tick +=1;

166

167 i f ( current Tick % ( 60 \* timeToCreatePowerUp )== 0 ) // kas 10 s s u k u r t i nauja powerUp

168 spawnPowerUp ( ) ;

169

170 f o r ( i n t x = 0 ; x < t h i s . g r i d S i z e ; x++)

171 {

172 f o r ( i n t y = 0 ; y < t h i s . g r i d S i z e ; y++)

173 {

174 i f ( t h i s . o b j e c t s [ x ] [ y ] . is Dead ) {

175

176

177

178 i f ( t h i s . o b j e c t s [ x ] [ y ] i n s t a n c e o f PowerUp)

179 {

180 System . out . p r i n t l n ( " Sunaikinau powerUpa " ) ;

181 powerUpCounter - =1 ;

182 }

183

184 t h i s . Clear Target ( x , y ) ;

185

186

187

188 }

189

190 o b j e c t s [ x ] [ y ] . onTick ( ) ;

191 }

192

}

}

p u b l i c i n t c e l l S i z e ( )

{

return s i z e / g r i d S i z e ;

}

}

193

194

195

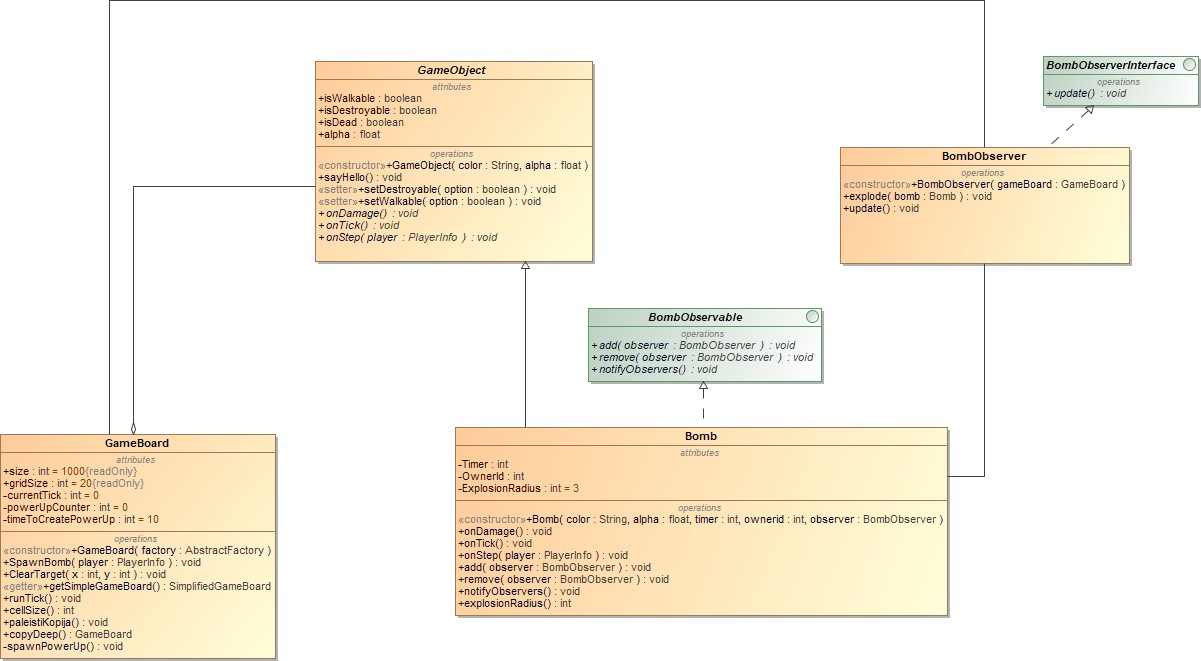
196

197

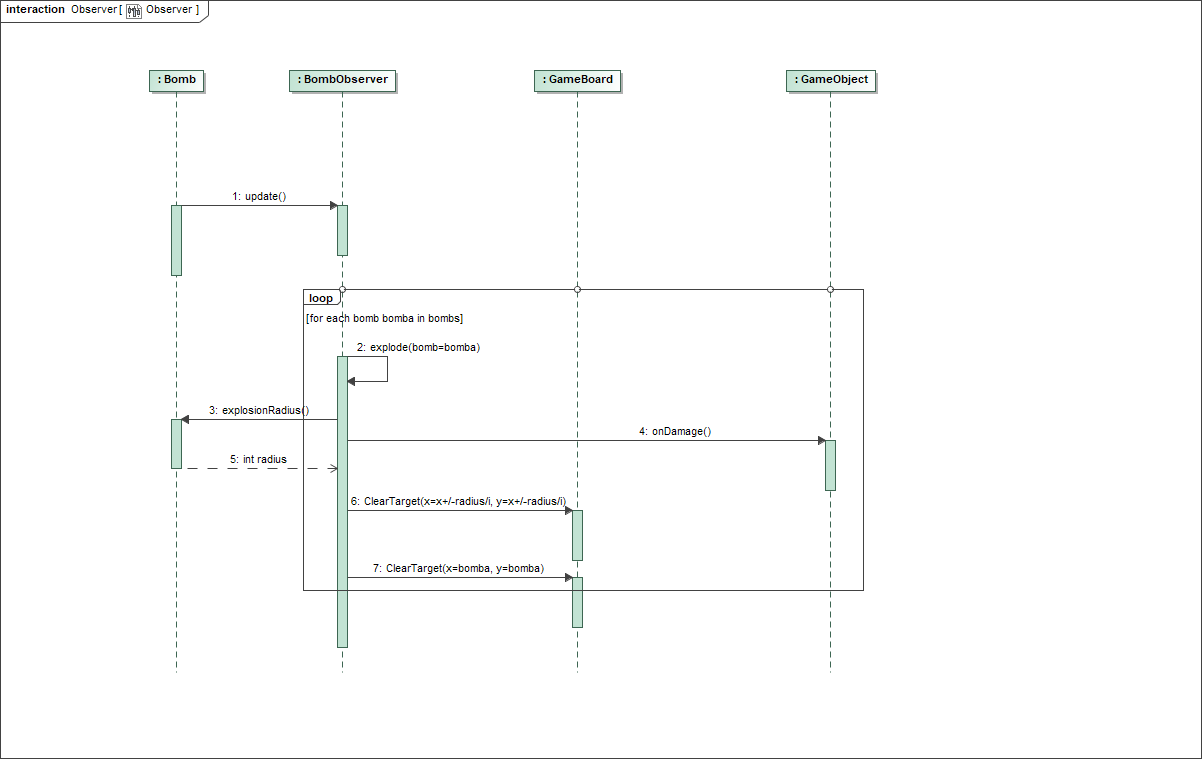
198

199

## Observer



### pav. Observer diagrama



* 1. **pav. Observer usecase**

Observer optimaliai apdoroja bombos informacija tik tada, kai bomba yra pasiruošusi sprogti.

package s e r v e r ;

import java . u t i l . L i s t ; import java . u t i l . Stack ;

p u b l i c c l a s s Bomb extends GameObject implements BombObservable{

p r i v a te i n t Timer ; p r i v a te i n t OwnerId ;

p r i v a te i n t Explosion Radius = 3 ;

p r i v a te List <BombObserver> o b s e r v e r s = new Stack<BombObserver >() ;

1

2

3

4

5

6

7

8

9

10

11

12

13 p u b l i c Bomb( S tr i n g color , f l o a t alpha , i n t timer , i n t ownerid , BombObserver o b s e r v e r ) {

14 super ( c o l or , alpha ) ;

15 t h i s . Timer = timer ;

16 t h i s . OwnerId = ownerid ;

17 t h i s . add ( o b s e r v e r ) ;

18 t h i s . i s Walkable = true ;

19 }

20

21 p u b l i c void onDamage ( ) {

22 // cannot be destroyed ?

23 }

24 p u b l i c void onTick ( ) {

25 t h i s . Timer - - ;

26 i f ( t h i s . Timer <= 0 )

27 n o t i f y O b s e r v e r s ( ) ;

28 }

29

30 p u b l i c void onStep ( Pl a y e r In f o p l a y e r ) {

31 // cannot be stepped on?

32 }

33

34 p u b l i c void add ( BombObserver o b s e rv e r )

35 {

36 t h i s . o b s e r v e r s . add ( ob s e r v e r ) ;

37 }

38

39 p u b l i c void remove ( BombObserver o b s e r v e r )

40 {

41

t h i s . o b s e r v e r s . remove ( o b s e r ve r ) ;

}

p u b l i c void n o t i f y O b s e r v e r s ( )

{

f o r ( BombObserver o b s e r v e r : o b s e r v e r s ) {

o b s e r ve r . explode ( t h i s ) ;

}

}

p u b l i c i n t explosion Radius ( ) {

i f ( t h i s . Timer <= 0 ) return t h i s . Explosion Radius ; e l s e return 0 ;

}

}

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

|  |  |  |  |
| --- | --- | --- | --- |
| 1 package s e r v e r ;  2  3 p u b l i c i n t e r f a c e BombObservable { | | | |
| 4 | p u b l i c | void | add ( BombObserver o b s e r v e r ) ; |
| 5 | p u b l i c | void | remove ( BombObserver o b s e rv e r ) ; |
| 6  7  8 } | p u b l i c | void | n o t i f y O b s e r v e r s ( ) ; |

1

package s e r v e r ;

import java . u t i l . L i s t ;

2

3

4 import java . u t i l . Stack ;

5

6 p u b l i c c l a s s BombObserver implements Bomb Observer Interface {

7 p r i v a te GameBoard gameBoard ;

8 p r i v a te List <Bomb> bombs = new Stack<Bomb>() ;

9

10

11 p u b l i c BombObserver ( GameBoard gameBoard ) {

12 t h i s . gameBoard = gameBoard ;

13 }

14

15 p u b l i c void update ( ) {

16 f o r ( i n t x = 0 ; x < t h i s . gameBoard . g r i d S i z e ; x++) {

17 f o r ( i n t y = 0 ; y < t h i s . gameBoard . g r i d S i z e ; y++) {

18 i f ( t h i s . gameBoard . o b j e c t s [ x ] [ y ] . g e t C l a s s ( ) . e q u a l s (Bomb

. c l a s s ) )

19 {

20 bombs . add ( ( Bomb) t h i s . gameBoard . o b j e c t s [ x ] [ y ] ) ;

21 }

22 }

23 }

24

25

26

27

28

29

30 }

f o r (Bomb bomb : bombs ) {

i f (bomb . explosion Radius ( ) > 0 ) { explode (bomb) ;

}

}

31

|  |  |  |  |
| --- | --- | --- | --- |
| 32 | p u b l i c void | | explode (Bomb bomb) { |
| 33 | f o r ( i n t | | x = 0 ; x < t h i s . gameBoard . g r i d S i z e ; x++) |
| 34 | { | |  |
| 35 | f o r | | ( i n t y = 0 ; y < t h i s . gameBoard . g r i d S i z e ; y++) |
| 36 | { | |  |
| 37 |  | | i f ( t h i s . gameBoard . o b j e c t s [ x ] [ y ] == bomb) |
| 38 |  | | { |
| 39 |  | | f o r ( i n t i = 0 ; i < bomb . explosion Radius ( ) ; i++) { |
| 40  41 | c a l c u l a t i o n | | //TODO: Make a s o p h i s t i c a t e d e x p l o s i o n r a d i u s  t h i s . gameBoard . Clear Target ( x , y ) ; |
| 42 |  | | i f ( x + i < t h i s . gameBoard . g r i d S i z e ) { |
| 43 | ; | | t h i s . gameBoard . o b j e c t s [ x + i ] [ y ] . onDamage ( ) |
| 44  45 | ) | i f ( t h i s . gameBoard . o b j e c t s [ x + i ] [ y ] . is Dead t h i s . gameBoard . Clear Target ( x + 1 , y ) ;  } | |
| 46 |  | i f ( y + i < t h i s . gameBoard . g r i d S i z e ) { | |
| 47 |  | t h i s . gameBoard . o b j e c t s [ x ] [ y + i ] . onDamage ( ) | |
| 48  49  50  51  52  53 | ;  i f ( t h i s . gameBoard . o b j e c t s [ x ] [ y + i ] . is Dead  ) t h i s . gameBoard . Clear Target ( x , y + 1 ) ;  }  i f ( x - i > 0 ) {  t h i s . gameBoard . o b j e c t s [ x - i ] [ y ] . onDamage ( )  ;  i f ( t h i s . gameBoard . o b j e c t s [ x - i ] [ y ] . is Dead  ) t h i s . gameBoard . Clear Target ( x - 1 , y ) ;  } | | |

54

i f ( y - i > 0 ) {

t h i s . gameBoard . o b j e c t s [ x ] [ y - i ] . onDamage ( )

;

i f ( t h i s . gameBoard . o b j e c t s [ x ] [ y - i ] . is Dead

55

56

|  |  |  |
| --- | --- | --- |
| 57 | ) t h i s . gameBoard . Clear Target ( x , y -  } | 1 ) ; |
| 58 | } |  |
| 59 | } |  |
| 60 | } |  |
| 61 | } |  |
| 62 | } |  |
| 63 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c i n t e r f a c e Bomb Observer Interface | { |
| 4 | p u b l i c void update ( ) ; |  |
| 5 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c a b s tr a c t c l a s s GameObject { |  |
| 4 | p u b l i c boolean i s Walkable ; |  |
| 5 | p u b l i c boolean i s D e s tr o y a b l e ; |  |
| 6 | p u b l i c boolean is Dead ; |  |
| 7 |  |  |
| 8  9  10  11 | GameObjectDelegate gameobjectdelegate ;  p u b l i c S tr i n g c o l o r ; p u b l i c f l o a t alpha ; | |

12

13 p u b l i c GameObject ( S tr i n g color , f l o a t alpha ) {

14 super ( ) ;

15 t h i s . alpha = alpha ;

16 t h i s . c o l o r = c o l o r ;

17 }

18

19 p u b l i c void say Hello ( ) {

20 System . out . p r i n t l n ( " GameObject " ) ;

21 }

22

23 p u b l i c void s e t D e s tr o y a b l e ( boolean option ) {

24 t h i s . i s D e s tr o y a b l e = option ;

25 }

26

27 p u b l i c void set Walkable ( boolean option ) {

28 t h i s . i s Walkable = option ;

29 }

30

31 p u b l i c a b s tr a c t void onDamage ( ) ;

32 p u b l i c a b s tr a c t void onTick ( ) ;

33 p u b l i c a b s tr a c t void onStep ( Pl a y e r In f o p l a y e r ) ;

34

35 }

1

package s e r v e r ;

import java . u t i l . Array List ; import java . u t i l . Random ;

2

3

4

5

6 import shared . \* ;

7

8 p u b l i c c l a s s GameBoard implements Cloneable {

9

10 p u b l i c f i n a l i n t s i z e = 1000 ;

11 p u b l i c f i n a l i n t g r i d S i z e = 2 0 ;

12 p u b l i c GameObject [ ] [ ] o b j e c t s ;

13 p r i v a te BombObserver bombObserver ;

14 p r i v a te Abstract Factory f a c to r y ;

15

16 p r i v a te i n t current Tick = 0 ;

17 p r i v a te i n t powerUpCounter = 0 ;

18 p r i v a te i n t timeToCreatePowerUp = 1 0 ;

19

20 p r i v a te ISta g e Bu i l d e r s ta g e 1 b u i l d e r ;

21 p r i v a te ISta g e Bu i l d e r s ta g e 2 b u i l d e r ;

22 p r i v a te ISta g e Bu i l d e r s ta g e 3 b u i l d e r ;

23

24 p u b l i c GameBoard( Abstract Factory f a c to r y )

25 {

26 t h i s . f a c to r y = f a c to r y ;

27 t h i s . o b j e c t s = new GameObject [ g r i d S i z e ] [ g r i d S i z e ] ;

28 t h i s . s ta g e 1 b u i l d e r = new Stage 1 Builder ( g r i d S i z e ) ;

29 t h i s . s ta g e 2 b u i l d e r = new Stage 2 Builder ( g r i d S i z e ) ;

30 t h i s . s ta g e 3 b u i l d e r = new Stage 3 Builder ( g r i d S i z e ) ;

31

32 Sta g e D i r e c to r s ta g e D i r e c to r = new Sta g e D i r e c to r ( s ta g e 3 b u i l d e r ) ;

33

34 s ta g e D i r e c to r . makeStage ( ) ;

35

36 Stage s tage 1 = s ta g e D i r e c to r . get Stage ( ) ;

37

38 f o r ( Coordinates kor : s tage 1 . getGrounds ( ) )

39 t h i s . o b j e c t s [ kor . getX ( ) ] [ kor . getY ( ) ] = t h i s . f a c to r y . create Ground ( ) ;

40

41 i n t k i e k i s = 7 6 ;

42 boolean sunaikinama = f a l s e ;

43

44 f o r ( Coordinates kor : s tage 1 . get Walls ( ) ) {

45

46 t h i s . o b j e c t s [ kor . getX ( ) ] [ kor . getY ( ) ] = t h i s . f a c to r y . create Wall ( sunaikinama ) ;

47 k i e k i s - =1 ;

48 i f ( k i e k i s == 0 ) sunaikinama = true ;

49 }

50

51

52

53 bombObserver = new BombObserver ( t h i s ) ;

54

55

56 t h i s . f a c to r y . SetBombObserver ( bombObserver ) ;

57

58

59

60

61

62 // t e s t

63

64 //TODO wrong but ok f o r now

65 t h i s . o b j e c t s [ 1 7 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

66

67 t h i s . o b j e c t s [ 1 5 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

68 Trap modified = ( Trap ) t h i s . o b j e c t s [ 1 5 ] [ 3 ] ;

69 modified . s e t T r a p e f f e c t ( new DamageTrap ( new Concrete Trap ( ) ) ) ;

70 t h i s . o b j e c t s [ 1 5 ] [ 3 ] = modified ;

71

72 t h i s . o b j e c t s [ 1 3 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

73 Trap modified 1 = ( Trap ) t h i s . o b j e c t s [ 1 3 ] [ 3 ] ;

74 modified 1 . s e t T r a p e f f e c t ( new TeleportTrap ( new Concrete Trap ( ) ) ) ;

75 t h i s . o b j e c t s [ 1 3 ] [ 3 ] = modified 1 ;

76

77 t h i s . o b j e c t s [ 1 1 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

78 Trap modified 2 = ( Trap ) t h i s . o b j e c t s [ 1 1 ] [ 3 ] ;

79 modified 2 . s e t T r a p e f f e c t ( new SlowTrap ( new Concrete Trap ( ) ) ) ;

80 t h i s . o b j e c t s [ 1 1 ] [ 3 ] = modified 2 ;

81

82 t h i s . o b j e c t s [ 9 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

83 Trap modified 3 = ( Trap ) t h i s . o b j e c t s [ 9 ] [ 3 ] ;

84 modified 3 . s e t T r a p e f f e c t ( new DamageTrap ( new TeleportTrap ( new SlowTrap ( new Concrete Trap ( ) ) ) ) ) ;

85 t h i s . o b j e c t s [ 9 ] [ 3 ] = modified 3 ;

86

87 }

88

89 p u b l i c void p a l e i s t i K o p i j a ( ) {

90 GameBoard copy = copyDeep ( ) ;

91

92

93 }

94

95

96

97 p u b l i c GameBoard copyDeep ( ) {

98

99 try {

100 return ( GameBoard) t h i s . c l o n e ( ) ;

101 } catch ( Clone NotSupported Exception e ) {

102 e . print Stack Trace ( ) ;

103 return n u l l ;

104 }

105 }

106

107 p u b l i c void SpawnBomb( Pl a y e r In f o p l a y e r )

108 {

109 i n t x = Math . round ( p l a y e r . c o o r d i n a te . x /( s i z e / g r i d S i z e ) ) ;

110 i n t y = Math . round ( p l a y e r . c o o r d i n a te . y /( s i z e / g r i d S i z e ) ) ;

111 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . createBomb ( p l a y e r . id ) ;

112

113 }

114

115 p u b l i c void Clear Target ( i n t x , i n t y ) {

116 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . create Ground ( ) ;

117 System . out . p r i n t l n ( " Removing bomb from l o c a t i o n " + x + " " + y

) ;

118 }

119

120 p u b l i c Simplified Game Board getSimpleGameBoard ( )

121 {

122 Simplified Game Board simpleGameboard = new Simplified Game Board ( t h i s . s i z e , t h i s . g r i d S i z e ) ;

123

124 f o r ( i n t i = 0 ; i < t h i s . g r i d S i z e ; i ++)

125 {

126 f o r ( i n t j = 0 ; j < t h i s . g r i d S i z e ; j ++)

127 {

128 simpleGameboard . o b j e c t s [ i ] [ j ] = new Simplified Game Object ( t h i s . o b j e c t s [ i ] [ j ] . col or , ObjectType .GROUND) ;

129 }

130 }

131 return simpleGameboard ;

132 }

133

134 p r i v a te void spawnPowerUp ( ) {

135 Random rand = new Random ( ) ;

136

137 i f ( powerUpCounter > 4 )

138 return ;

139

140 while ( true ) {

141 i n t x = rand . next Int (( 19 - 0 ) + 1 ) + 0 ;

142 i n t y = rand . next Int (( 19 - 0 ) + 1 ) + 0 ;

143

144

145 i f ( t h i s . o b j e c t s [ x ] [ y ] i n s t a n c e o f Ground ) {

146 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . createPowerUp ( ) ;

147 powerUpCounter+=1;

148 break ;

149 }

150

151 }

152

153

154

155

156

157

158

159

160 }

161

162

163 p u b l i c void run Tick ( )

164 {

165 current Tick +=1;

166

167 i f ( current Tick % ( 60 \* timeToCreatePowerUp )== 0 ) // kas 10 s s u k u r t i nauja powerUp

168 spawnPowerUp ( ) ;

169

170 f o r ( i n t x = 0 ; x < t h i s . g r i d S i z e ; x++)

171 {

172 f o r ( i n t y = 0 ; y < t h i s . g r i d S i z e ; y++)

173 {

174 i f ( t h i s . o b j e c t s [ x ] [ y ] . is Dead ) {

175

176

177

178 i f ( t h i s . o b j e c t s [ x ] [ y ] i n s t a n c e o f PowerUp)

179 {

180 System . out . p r i n t l n ( " Sunaikinau powerUpa " ) ;

181 powerUpCounter - =1 ;

182 }

183

184 t h i s . Clear Target ( x , y ) ;

185

186

187

188 }

189

190 o b j e c t s [ x ] [ y ] . onTick ( ) ;

191 }

192 }

193 }

194

195

196

197

198

199 }

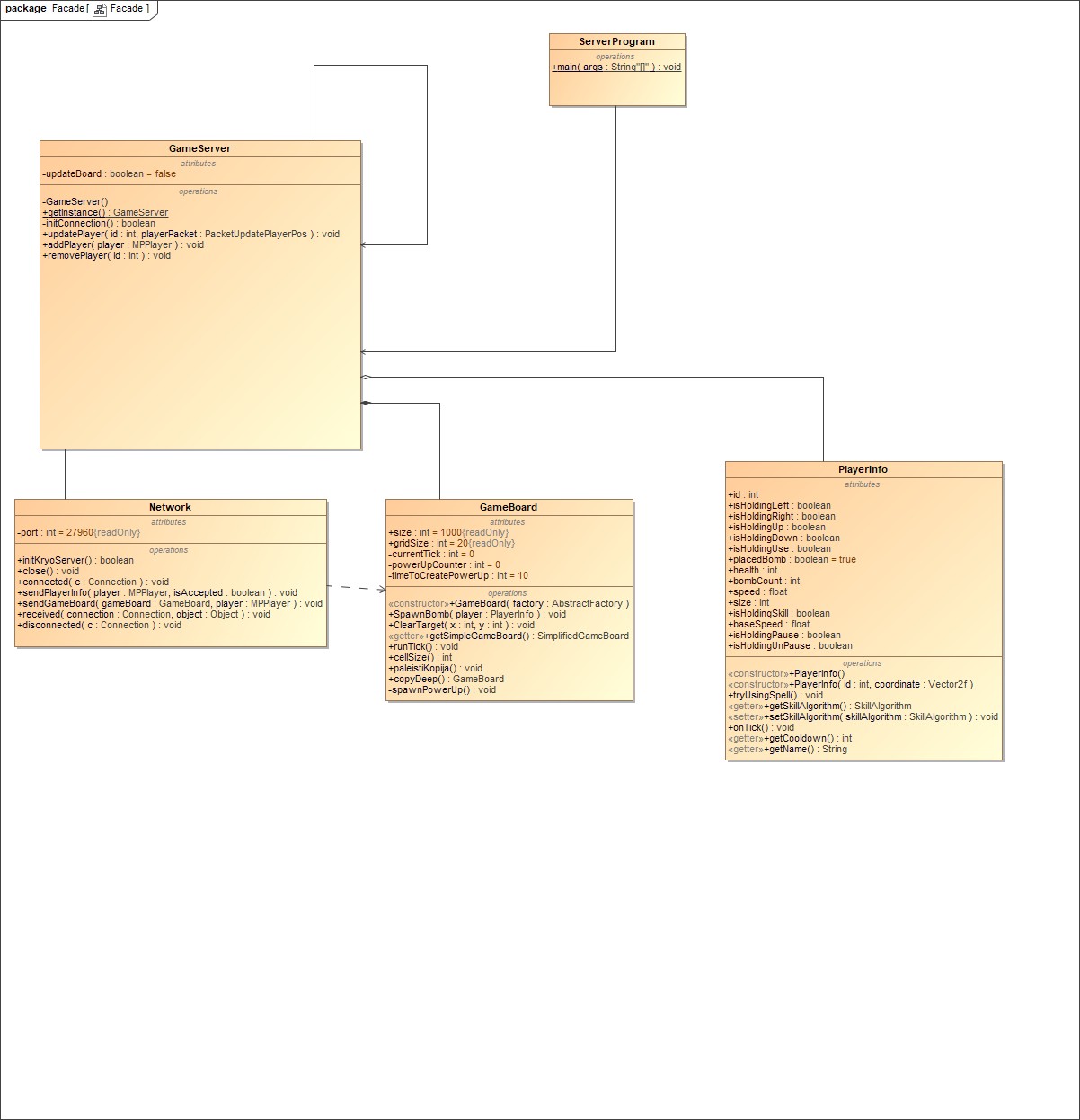
p u b l i c i n t c e l l S i z e ( )

{

return s i z e / g r i d S i z e ;

}

## Facade



### pav. Facade diagrama

Naudojant Facade mūsų serverio žaidimo logikos funkcionalumas yra prieinamas prie GameServer klasės kuri yra atitraukta nuo viso kito programos funkcionalumo, taigi jeigu reikėtų implemetuoti papil- domas sub sistemas, kaip monitoring arba serverio GUI, tai būtų lengviau.

|  |  |  |
| --- | --- | --- |
| 1  2  3 | package s e r v e r ;  p u b l i c c l a s s Server Program |  |
| 4 | { |
| 5 | p u b l i c s t a t i c void main ( S tr i n g [ ] | args ) |
| 6 | { |  |
| 7 | GameServer . g e t I n s ta n c e ( ) ; |  |
| 8 | } |  |
| 9 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | import java . u t i l . HashMap ; import java . u t i l . Map;  import shared . Vector 2 f ;  import shared . PacketUpdate Player Pos ;  c l a s s GameServer  {  p r i v a te s t a t i c GameServer game Server = n u l l ;  p r o te c te d v o l a t i l e GameBoard gameBoard ;  p r o te c te d v o l a t i l e Map<I nteger , MPPlayer> p l a y e r s ; p r o te c te d v o l a t i l e Network network ;  p r i v a te GameCycleThread thread ; p r i v a te Stage 1 Factory s ta g e 1 f a c to r y ; p r i v a te boolean updateBoard = f a l s e ; | |

20 p r i v a te GameServer ( )

21 {

22 // I n i t Connection

23 i f ( ! i n i t C o n n e c t i o n ( ) )

24 {

25 System . e r r . p r i n t l n ( "ERROR Connecting to host " ) ;

26 return ;

27 }

28

29 t h i s . p l a y e r s = new HashMap<Integer , MPPlayer >() ;

30 t h i s . s ta g e 1 f a c to r y = new Stage 1 Factory ( ) ;

31 t h i s . gameBoard = new GameBoard( s ta g e 1 f a c to r y ) ;

32

33

34

35 }

t h i s . thread = new GameCycleThread ( ) ; t h i s . thread . s t a r t ( ) ;

36

37 p u b l i c s t a t i c GameServer g e t I n s ta n c e ( )

38 {

39 i f ( game Server == n u l l )

40 {

41 game Server = new GameServer ( ) ;

42 }

43 return game Server ;

44 }

45

46 p r i v a te boolean i n i t C o n n e c t i o n ( )

47 {

48 t h i s . network = new Network ( ) ;

49

50 i f ( ! t h i s . network . i n i t Kr y o Se r v e r ( ) )

51 {

52 return f a l s e ;

53 }

54

55

56 }

return true ;

57

58 p u b l i c void update Player ( i n t id , PacketUpdate Player Pos player Packet )

59 {

60 MPPlayer p l a y e r = p l a y e r s . get ( id ) ;

61 i f ( p l a y e r != n u l l )

62 {

63 p l a y e r . is Holding Up = player Packet . is Holding Up != n u l l ? player Packet . is Holding Up : p l a y e r . is Holding Up ;

64 p l a y e r . isHoldingDown = player Packet . isHoldingDown != n u l l ? player Packet . isHoldingDown : p l a y e r . isHoldingDown ;

65 p l a y e r . i s H o l d i n g L e f t = player Packet . i s H o l d i n g L e f t != n u l l ? player Packet . i s H o l d i n g L e f t : p l a y e r . i s H o l d i n g L e f t ;

66 p l a y e r . i s Holding Right = player Packet . i s Holding Right != n u l l ? player Packet . i s Holding Right : p l a y e r . i s Holding Right ;

67 p l a y e r . i s Holding Use = player Packet . i s Holding Use != n u l l ? player Packet . i s Holding Use : p l a y e r . i s Holding Use ;

68 p l a y e r . i s H o l d i n g S k i l l = player Packet . i s H o l d i n g S k i l l != n u l l ? player Packet . i s H o l d i n g S k i l l : p l a y e r . i s H o l d i n g S k i l l ;

69 p l a y e r s . put ( p l a y e r . id , p l a y e r ) ;

70 }

71 }

72

73 p u b l i c void add Player ( MPPlayer p l a y e r )

74 {

75 t h i s . p l a y e r s . put ( p l a y e r . c . getID ( ) , p l a y e r ) ;

76 t h i s . network . sendGameBoard ( gameBoard , p l a y e r ) ;

77

78 }

79

80 p u b l i c void remove Player ( i n t id )

81 {

82

83

84 }

p l a y e r s . remove ( id ) ;

85

86

87 p r i v a te c l a s s GameCycleThread extends Thread

88 {

89 v o l a t i l e boolean isGameRunning = true ;

90 p r i v a te f i n a l i n t gameSpeed = 1 6 ; //The lower the number the f a s t e r the game i s

91

92 p u b l i c GameCycleThread ( )

93 {

94 t h i s . isGameRunning = true ;

95 }

96

97 p u b l i c void run ( )

98 {

99 while ( t h i s . isGameRunning )

100 {

101 try

102 {

103 // Probably should use Timer i n s te a d

104 Thread . s l e e p ( gameSpeed ) ;

105 t h i s . update ( ) ;

106 }

107 catch ( Interrupted Exception e )

108 {

109 e . print Stack Trace ( ) ;

110 t h i s . stopGame ( ) ;

111 }

112 }

113 }

114

115 p u b l i c void stopGame ( )

116 {

117 t h i s . isGameRunning = f a l s e ;

118

119 // network should probably be c l o s e d by the parent

120 network . c l o s e ( ) ;

121 }

122

123 p r i v a te void update ( )

124 {

125 update Players ( ) ;

126 gameBoard . run Tick ( ) ;

127 }

128

129 p r i v a te void update Players ( )

130 {

131 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

132 {

133 i f ( p . i s Holding Pause )

134 {

135

136 }

137

138 i f ( p . i s H o l d i n g S k i l l )

139 {

140 p . tr y U s i n g S p e l l ( ) ;

141 }

142

143 p . onTick ( ) ;

144

145 i f ( p . i s Holding Use )

146 {

147 gameBoard . SpawnBomb( p ) ;

148 }

149

150 p . c o o r d i n a te = c h e c k C o l l i s i o n ( p ) ;

151

152 network . sendGameBoard ( gameBoard , p ) ;

153 network . se n d Player In f o ( p , true ) ;

154 }

155 }

156

157 p r i v a te Vector 2 f c h e c k C o l l i s i o n ( MPPlayer p )

158 {

159

160 Vector 2 f coords After Move = new Vector 2 f ( p . c o o r d i n a te . x , p . c o o r d i n a te . y ) ;

161

162 f l o a t padding = 0 . 001 f ;

163 f l o a t c e l l S i z e = gameBoard . c e l l S i z e ( ) ;

164

165 boolean moveX = true ;

166 boolean moveY = true ;

167

168 i f ( p . i s H o l d i n g L e f t )

169 {

170 coords After Move . x -= p . speed ;

171 }

172

173 i f ( p . i s Holding Right )

174 {

175

176 coords After Move . x += p . speed ;

177 }

178

179 boolean c o l l i d i n g L e f t = ( ( i n t ) coords After Move . x / c e l l S i z e - padding ) < ( ( i n t ) p . c o o r d i n a te . x / c e l l S i z e ) ;

180 boolean c o l l i d i n g R i g h t = ( ( ( i n t ) coords After Move . x + p . s i z e + padding ) / c e l l S i z e ) > ( ( ( i n t ) p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

181 boolean i s C o l l i d i n g X = c o l l i d i n g L e f t | | c o l l i d i n g R i g h t ;

182 moveX = ! ( coords After Move . x <= 0 | | coords After Move . x >= gameBoard . s i z e - p . s i z e ) ;

183 //Some smoothing when going around edges would be n i c e

184 i f ( i s C o l l i d i n g X && moveX)

185 {

186 i n t x = 0 , y = 0 , y1 = 0 ;

187 i f ( c o l l i d i n g L e f t )

188 {

189 x = ( i n t ) ( ( coords After Move . x ) / c e l l S i z e ) ;

190 y = ( i n t ) ( p . c o o r d i n a te . y / c e l l S i z e ) ;

191 y1 = ( i n t ) ( ( p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

192

193 }

194 i f ( c o l l i d i n g R i g h t )

195 {

196 x = ( i n t ) ( ( coords After Move . x + p . s i z e ) / c e l l S i z e ) ;

197 y = ( i n t ) ( p . c o o r d i n a te . y / c e l l S i z e ) ;

198 y1 = ( i n t ) ( ( p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

199

200 }

201

202 i f ( y == y1 )

203 {

204 moveX = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable ;

205 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

206 }

207 e l s e

208 {

209 moveX = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable && gameBoard . o b j e c t s [ x ] [ y1 ] . i s Walkable ;

210 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

211 gameBoard . o b j e c t s [ x ] [ y1 ] . onStep ( p ) ;

212 }

213 }

214

215

216 i f ( p . is Holding Up )

217 {

218 coords After Move . y += p . speed ;

219 }

220

221 i f ( p . isHoldingDown )

222 {

223 coords After Move . y -= p . speed ;

224 }

225

226 boolean c o l l i d i n g U p = ( ( ( i n t ) coords After Move . y + p . s i z e - padding

) / c e l l S i z e ) > ( ( ( i n t ) p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

227 boolean colliding Down = ( ( i n t ) coords After Move . y / c e l l S i z e + padding ) < ( ( i n t ) p . c o o r d i n a te . y / c e l l S i z e ) ;

228 boolean i s C o l l i d i n g Y = c o l l i d i n g U p | | colliding Down ;

229 moveY = ! ( coords After Move . y <= 0 | | coords After Move . y >= gameBoard . s i z e - p . s i z e ) ;

230

231 //Some smoothing when going around edges would be n i c e

232 i f ( i s C o l l i d i n g Y && moveY)

233 {

234 i n t x = 0 , x1 = 0 , y = 0 ;

235

236 i f ( c o l l i d i n g U p )

237 {

238 y = ( i n t ) ( ( coords After Move . y + p . s i z e ) / c e l l S i z e ) ;

239 x = ( i n t ) ( p . c o o r d i n a te . x / c e l l S i z e ) ;

240 x1 = ( i n t ) ( ( p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

241

242 }

243 i f ( colliding Down )

244 {

245 y = ( i n t ) ( ( coords After Move . y ) / c e l l S i z e ) ;

246 x = ( i n t ) ( p . c o o r d i n a te . x / c e l l S i z e ) ;

247 x1 = ( i n t ) ( ( p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

248 }

249

250 i f ( x == x1 )

251 {

252 moveY = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable ;

253 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

254 }

255 e l s e

256 {

257 moveY = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable && gameBoard . o b j e c t s [ x1 ] [ y ] . i s Walkable ;

258 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

259 gameBoard . o b j e c t s [ x1 ] [ y ] . onStep ( p ) ;

260 }

261 }

262

263 coords After Move . x = moveX ? coords After Move . x : p . c o o r d i n a te . x ;

264 coords After Move . y = moveY ? coords After Move . y : p . c o o r d i n a te . y ;

265

return coords After Move ;

}

}

}

266

267

268

269

270

271

272

1

package s e r v e r ;

import org . l w j g l . system . Callback I . P;

import com . e s o t e r i c s o f t w a r e . kryonet . Connection ; import com . e s o t e r i c s o f t w a r e . kryonet . L i s te n e r ; import com . e s o t e r i c s o f t w a r e . kryonet . Server ;

import shared . PacketUpdate Player Pos ; import shared . Simplified Game Board ; import shared . Simplified Game Object ; import shared . ObjectType ;

import shared . PacketAddPlayer ; import shared . PacketRemovePlayer ; import shared . PacketUpdateGameBoard ; import shared . Vector 2 f ;

p u b l i c c l a s s Network extends L i s te n e r

{

p r i v a te Server s e r v e r ;

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21 p r i v a te f i n a l i n t port = 27960 ;

22

23 p u b l i c boolean i n i t Kr y o Se r v e r ( )

24 {

25 try

26 {

27 t h i s . s e r v e r = new Server ( 131072 , 16384 ) ;

28 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( PacketUpdate Player Pos . c l a s s ) ;

29 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( PacketAddPlayer . c l a s s ) ;

30 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( PacketRemovePlayer . c l a s s ) ;

31 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( PacketUpdateGameBoard . c l a s s ) ;

32 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( Simplified Game Board . c l a s s ) ;

33 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( ObjectType . c l a s s ) ;

34 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( Simplified Game Object . c l a s s ) ;

35 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( Simplified Game Object [ ] . c l a s s ) ;

36 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( Simplified Game Object [ ] [ ] . c l a s s ) ;

37 t h i s . s e r v e r . getKryo ( ) . r e g i s t e r ( Vector 2 f . c l a s s ) ;

38 t h i s . s e r v e r . bind ( t h i s . port , t h i s . port ) ;

39 t h i s . s e r v e r . add Listener ( t h i s ) ;

40 t h i s . s e r v e r . s t a r t ( ) ;

41 System . out . p r i n t l n ( " The s e r v e r i s ready " ) ;

42 return true ;

43 }

44 catch ( Exception e )

45 {

46 e . print Stack Trace ( ) ;

47 return f a l s e ;

48 }

49 }

50

51 p u b l i c void c l o s e ( )

52 {

53 t h i s . s e r v e r . c l o s e ( ) ;

54 }

55

56 p u b l i c void connected ( Connection c )

57 {

58 MPPlayer p l a y e r = new MPPlayer ( ) ;

59 p l a y e r . c = c ;

60

61

62

63

64

65

66 }

PacketAddPlayer packet = new PacketAddPlayer ( ) ; packet . id = c . getID ( ) ;

t h i s . s e r v e r . sendToAllExceptTCP ( c . getID ( ) , packet ) ; GameServer . g e t I n s ta n c e ( ) . add Player ( p l a y e r ) ;

System . out . p r i n t l n ( " Connection r e c e i v e d . " ) ;

67

68 p u b l i c void s e nd Pl aye r In f o ( MPPlayer player , boolean i s Accepted )

69 {

70 PacketUpdate Player Pos packet = new PacketUpdate Player Pos ( p l a y e r . c . getID ( ) , p l a y e r ) ;

71 packet . accepted = i s Accepted ;

72 t h i s . s e r v e r . sendToAllUDP ( packet ) ;

73

74 }

75

76 p u b l i c void sendGameBoard ( GameBoard gameBoard , MPPlayer p l a y e r )

77 {

78

79

80

81

82

83

84

85

86

87

88

89 }

Simplified Game Board simpleGameboard = gameBoard . getSimpleGameBoard ( ) ;

i f ( p l a y e r == n u l l )

{

PacketUpdateGameBoard packet = new PacketUpdateGameBoard ( simpleGameboard ) ;

t h i s . s e r v e r . sendToAllUDP ( packet ) ;

}

e l s e

{

PacketUpdateGameBoard packet = new PacketUpdateGameBoard ( simpleGameboard ) ;

t h i s . s e r v e r . sendToUDP( p l a y e r . c . getID ( ) , packet ) ;

}

90

91 p u b l i c void r e c e i v e d ( Connection connection , Object o b je c t )

92 {

93 i f ( o b je c t i n s t a n c e o f PacketUpdate Player Pos )

94 {

95 PacketUpdate Player Pos packet = ( PacketUpdate Player Pos ) o b je c t ;

96 GameServer . g e t I n s ta n c e ( ) . update Player ( connection . getID ( ) , packet )

;

97 System . out . p r i n t l n ( " Received c o o r d i n a te packet " ) ;

98 }

99 }

100

101 p u b l i c void disconnected ( Connection c )

102 {

103

PacketRemovePlayer packet = new PacketRemovePlayer ( ) ; packet . id = c . getID ( ) ;

t h i s . s e r v e r . sendToAllExceptTCP ( c . getID ( ) , packet ) ; GameServer . g e t I n s ta n c e ( ) . remove Player ( c . getID ( ) ) ; System . out . p r i n t l n ( " Connection dropped . " ) ;

}

}

104

105

106

107

108

109

1

package s e r v e r ;

import java . u t i l . Array List ; import java . u t i l . Random ;

import shared . \* ;

p u b l i c c l a s s GameBoard implements Cloneable { p u b l i c f i n a l i n t s i z e = 1000 ;

p u b l i c f i n a l i n t g r i d S i z e = 2 0 ; p u b l i c GameObject [ ] [ ] o b j e c t s ; p r i v a te BombObserver bombObserver ; p r i v a te Abstract Factory f a c to r y ;

p r i v a te i n t current Tick = 0 ; p r i v a te i n t powerUpCounter = 0 ;

p r i v a te i n t timeToCreatePowerUp = 1 0 ;

p r i v a te ISta g e Bu i l d e r s ta g e 1 b u i l d e r ; p r i v a te ISta g e Bu i l d e r s ta g e 2 b u i l d e r ;

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22 p r i v a te ISta g e Bu i l d e r s ta g e 3 b u i l d e r ;

23

24 p u b l i c GameBoard( Abstract Factory f a c to r y )

25 {

26 t h i s . f a c to r y = f a c to r y ;

27 t h i s . o b j e c t s = new GameObject [ g r i d S i z e ] [ g r i d S i z e ] ;

28 t h i s . s ta g e 1 b u i l d e r = new Stage 1 Builder ( g r i d S i z e ) ;

29 t h i s . s ta g e 2 b u i l d e r = new Stage 2 Builder ( g r i d S i z e ) ;

30 t h i s . s ta g e 3 b u i l d e r = new Stage 3 Builder ( g r i d S i z e ) ;

31

32 Sta g e D i r e c to r s ta g e D i r e c to r = new Sta g e D i r e c to r ( s ta g e 3 b u i l d e r ) ;

33

34 s ta g e D i r e c to r . makeStage ( ) ;

35

36 Stage s tage 1 = s ta g e D i r e c to r . get Stage ( ) ;

37

38 f o r ( Coordinates kor : s tage 1 . getGrounds ( ) )

39 t h i s . o b j e c t s [ kor . getX ( ) ] [ kor . getY ( ) ] = t h i s . f a c to r y . create Ground ( ) ;

40

41 i n t k i e k i s = 7 6 ;

42 boolean sunaikinama = f a l s e ;

43

44 f o r ( Coordinates kor : s tage 1 . get Walls ( ) ) {

45

46 t h i s . o b j e c t s [ kor . getX ( ) ] [ kor . getY ( ) ] = t h i s . f a c to r y . create Wall ( sunaikinama ) ;

47 k i e k i s - =1 ;

48 i f ( k i e k i s == 0 ) sunaikinama = true ;

49 }

50

51

52

53 bombObserver = new BombObserver ( t h i s ) ;

54

55

56 t h i s . f a c to r y . SetBombObserver ( bombObserver ) ;

57

58

59

60

61

62 // t e s t

63

64 //TODO wrong but ok f o r now

65 t h i s . o b j e c t s [ 1 7 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

66

67 t h i s . o b j e c t s [ 1 5 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

68 Trap modified = ( Trap ) t h i s . o b j e c t s [ 1 5 ] [ 3 ] ;

69 modified . s e t T r a p e f f e c t ( new DamageTrap ( new Concrete Trap ( ) ) ) ;

70 t h i s . o b j e c t s [ 1 5 ] [ 3 ] = modified ;

71

72 t h i s . o b j e c t s [ 1 3 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

73 Trap modified 1 = ( Trap ) t h i s . o b j e c t s [ 1 3 ] [ 3 ] ;

74 modified 1 . s e t T r a p e f f e c t ( new TeleportTrap ( new Concrete Trap ( ) ) ) ;

75 t h i s . o b j e c t s [ 1 3 ] [ 3 ] = modified 1 ;

76

77 t h i s . o b j e c t s [ 1 1 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

78 Trap modified 2 = ( Trap ) t h i s . o b j e c t s [ 1 1 ] [ 3 ] ;

79 modified 2 . s e t T r a p e f f e c t ( new SlowTrap ( new Concrete Trap ( ) ) ) ;

80 t h i s . o b j e c t s [ 1 1 ] [ 3 ] = modified 2 ;

81

82 t h i s . o b j e c t s [ 9 ] [ 3 ] = t h i s . f a c to r y . create Trap ( ) ;

83 Trap modified 3 = ( Trap ) t h i s . o b j e c t s [ 9 ] [ 3 ] ;

84 modified 3 . s e t T r a p e f f e c t ( new DamageTrap ( new TeleportTrap ( new SlowTrap ( new Concrete Trap ( ) ) ) ) ) ;

85 t h i s . o b j e c t s [ 9 ] [ 3 ] = modified 3 ;

86

87 }

88

89 p u b l i c void p a l e i s t i K o p i j a ( ) {

90 GameBoard copy = copyDeep ( ) ;

91

92

93 }

94

95

96

97 p u b l i c GameBoard copyDeep ( ) {

98

99 try {

100 return ( GameBoard) t h i s . c l o n e ( ) ;

101 } catch ( Clone NotSupported Exception e ) {

102 e . print Stack Trace ( ) ;

103 return n u l l ;

104 }

105 }

106

107 p u b l i c void SpawnBomb( Pl a y e r In f o p l a y e r )

108 {

109 i n t x = Math . round ( p l a y e r . c o o r d i n a te . x /( s i z e / g r i d S i z e ) ) ;

110 i n t y = Math . round ( p l a y e r . c o o r d i n a te . y /( s i z e / g r i d S i z e ) ) ;

111 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . createBomb ( p l a y e r . id ) ;

112

113 }

114

115 p u b l i c void Clear Target ( i n t x , i n t y ) {

116 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . create Ground ( ) ;

117 System . out . p r i n t l n ( " Removing bomb from l o c a t i o n " + x + " " + y

) ;

118 }

119

120 p u b l i c Simplified Game Board getSimpleGameBoard ( )

121 {

122 Simplified Game Board simpleGameboard = new Simplified Game Board ( t h i s . s i z e , t h i s . g r i d S i z e ) ;

123

124 f o r ( i n t i = 0 ; i < t h i s . g r i d S i z e ; i ++)

125 {

126 f o r ( i n t j = 0 ; j < t h i s . g r i d S i z e ; j ++)

127 {

128 simpleGameboard . o b j e c t s [ i ] [ j ] = new Simplified Game Object ( t h i s . o b j e c t s [ i ] [ j ] . col or , ObjectType .GROUND) ;

129 }

130 }

131 return simpleGameboard ;

132 }

133

134 p r i v a te void spawnPowerUp ( ) {

135 Random rand = new Random ( ) ;

136

137 i f ( powerUpCounter > 4 )

138 return ;

139

140 while ( true ) {

141 i n t x = rand . next Int (( 19 - 0 ) + 1 ) + 0 ;

142 i n t y = rand . next Int (( 19 - 0 ) + 1 ) + 0 ;

143

144

145 i f ( t h i s . o b j e c t s [ x ] [ y ] i n s t a n c e o f Ground ) {

146 t h i s . o b j e c t s [ x ] [ y ] = t h i s . f a c to r y . createPowerUp ( ) ;

147 powerUpCounter+=1;

148 break ;

149 }

150

151 }

152

153

154

155

156

157

158

159

160 }

161

162

163 p u b l i c void run Tick ( )

164 {

165 current Tick +=1;

166

167 i f ( current Tick % ( 60 \* timeToCreatePowerUp )== 0 ) // kas 10 s s u k u r t i nauja powerUp

168 spawnPowerUp ( ) ;

169

170 f o r ( i n t x = 0 ; x < t h i s . g r i d S i z e ; x++)

171 {

172 f o r ( i n t y = 0 ; y < t h i s . g r i d S i z e ; y++)

173 {

174 i f ( t h i s . o b j e c t s [ x ] [ y ] . is Dead ) {

175

176

177

178 i f ( t h i s . o b j e c t s [ x ] [ y ] i n s t a n c e o f PowerUp)

179 {

180 System . out . p r i n t l n ( " Sunaikinau powerUpa " ) ;

181 powerUpCounter - =1 ;

182 }

183

184 t h i s . Clear Target ( x , y ) ;

185

186

187

188 }

189

o b j e c t s [ x ] [ y ] . onTick ( ) ;

}

}

}

p u b l i c i n t c e l l S i z e ( )

{

return s i z e / g r i d S i z e ;

}

}

190

191

192

193

194

195

196

197

198

199

1

package s e r v e r ;

import shared . Vector 2 f ; p u b l i c c l a s s Pl a y e r In f o

{

p u b l i c i n t id ;

p u b l i c Vector 2 f c o o r d i n a te ; p u b l i c boolean i s H o l d i n g L e f t ; p u b l i c boolean i s Holding Right ; p u b l i c boolean is Holding Up ; p u b l i c boolean isHoldingDown ; p u b l i c boolean i s Holding Use ; p u b l i c boolean i s H o l d i n g S k i l l ; p u b l i c boolean i s Holding Pause ; p u b l i c boolean is Holding Un Pause ;

p u b l i c boolean placedBomb = true ;

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18 p u b l i c i n t health ;

19 p u b l i c f l o a t speed ;

20 p u b l i c f l o a t base Speed ;

21 p u b l i c i n t s i z e ;

22 p u b l i c i n t bombCount ;

23

24 p r i v a te S k i l l A l g o r i th m s k i l l A l g o r i th m ;

25

26 p u b l i c Pl a y e r In f o ( )

27 {

28 t h i s . c o o r d i n a te = new Vector 2 f ( ) ;

29 t h i s . c o o r d i n a te . x = 400 ;

30 t h i s . c o o r d i n a te . y = 400 ;

31 t h i s . i s H o l d i n g L e f t = f a l s e ;

32 t h i s . i s Holding Right = f a l s e ;

33 t h i s . is Holding Up = f a l s e ;

34 t h i s . isHoldingDown = f a l s e ;

35 t h i s . i s Holding Pause = f a l s e ;

36 t h i s . is Holding Un Pause = f a l s e ;

37 t h i s . s k i l l A l g o r i th m = new Dash Skill ( ) ;

38 t h i s . s i z e = 4 0 ;

39 t h i s . speed = 2 . 5 f ;

40 t h i s . base Speed = 2 . 5 f ;

41 t h i s . health = 3 ;

42 t h i s . bombCount = 2 ;

43 // t h i s . p l a y e r S ta ts = new Concrete Player ( ) ;

44 }

45

46 p u b l i c Pl a y e r In f o ( i n t id , Vector 2 f c o o r d i n a te )

47 {

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63 }

t h i s . id = id ;

t h i s . c o o r d i n a te = c o o r d i n a te ; t h i s . i s H o l d i n g L e f t = f a l s e ; t h i s . i s Holding Right = f a l s e ; t h i s . is Holding Up = f a l s e ;

t h i s . isHoldingDown = f a l s e ; t h i s . i s Holding Use = f a l s e ; t h i s . i s Holding Pause = f a l s e ;

t h i s . is Holding Un Pause = f a l s e ; t h i s . s i z e = 4 0 ;

t h i s . speed = 2 . 5 f ;

t h i s . base Speed = 2 . 5 f ; t h i s . health = 3 ;

t h i s . bombCount = 2 ;

// Test

64

65 p u b l i c S k i l l A l g o r i th m g e t S k i l l A l g o r i th m ( )

66 {

67 return s k i l l A l g o r i th m ;

68 }

69

70 p u b l i c void s e t S k i l l A l g o r i th m ( S k i l l A l g o r i th m s k i l l A l g o r i th m )

71 {

72 t h i s . s k i l l A l g o r i th m = s k i l l A l g o r i th m ;

73 }

74

75

76

p u b l i c void onTick ( )

{

t h i s . s k i l l A l g o r i th m . onTick ( t h i s ) ;

}

p u b l i c void tr y U s i n g S p e l l ( )

{

t h i s . s k i l l A l g o r i th m . u s e S k i l l ( t h i s ) ;

}

p u b l i c i n t getCooldown ( )

{

return t h i s . s k i l l A l g o r i th m . getCooldown ( ) ;

}

p u b l i c S tr i n g getName ( )

{

return t h i s . s k i l l A l g o r i th m . getName ( ) ;

}

}

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

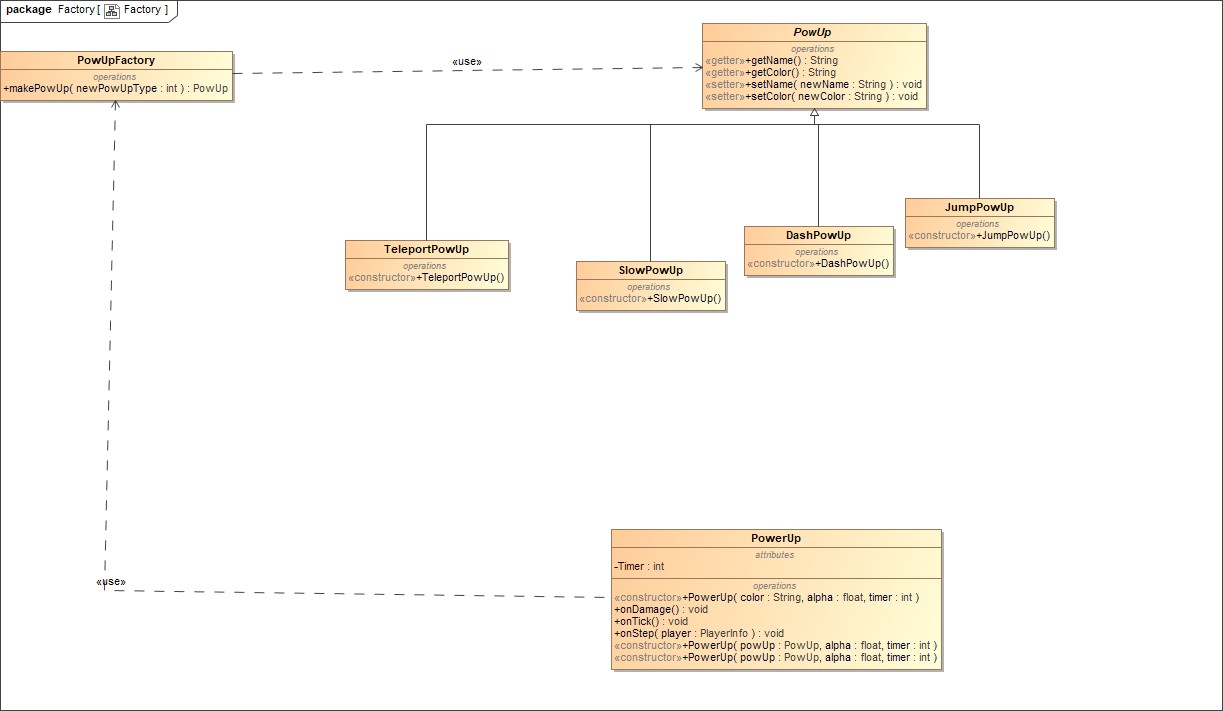
92

93

94

95

## Factory



### pav. Factory diagrama

Factory šablono poreikis atsirado norint kad dinamiškai keistusį žaidėjo įgudžiai.

package s e r v e r ;

p u b l i c c l a s s PowUpFactory {

p u b l i c PowUp makePowUp( i n t newPowUpType) { PowUp newPowUp = n u l l ;

i f ( 1 == newPowUpType) return new JumpPowUp() ;

e l s e i f ( 2 == newPowUpType) return new DashPowUp ( ) ;

1

2

3

4

5

6

7

8

9

10

11

12

e l s e i f ( 3 == newPowUpType)

return new SlowPowUp ( ) ;

e l s e i f ( 4 == newPowUpType)

return new TeleportPowUp ( ) ;

return newPowUp ;

}

}

13

14

15

16

17

18

19

20

21

22

23

1

package s e r v e r ;

p u b l i c a b s tr a c t c l a s s PowUp {

p r i v a te S tr i n g name ; p r i v a te S tr i n g c o l o r ;

p u b l i c S tr i n g getName ( ) { return name ;

}

p u b l i c S tr i n g get Color ( ) { return c o l o r ;

}

2

3

4

5

6

7

8

9

10

11

12

13

14

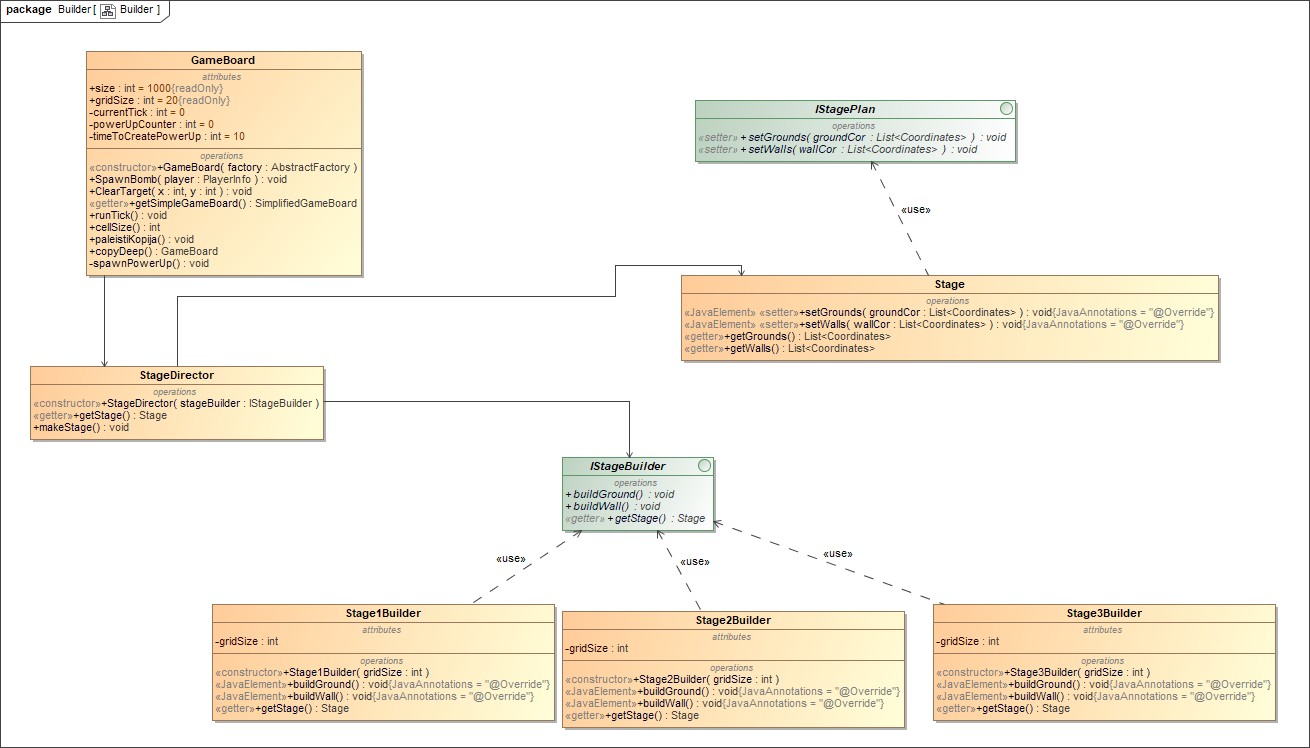
15

16

|  |  |  |  |
| --- | --- | --- | --- |
| 17 |  | | |
| 18 | p u | b l i c void setName ( S tr i n g newName) { | |
| 19 |  | name = newName ; | |
| 20 | } |  | |
| 21 |  |  | |
| 22  23  24 | p u b l i c void s e t C o l o r ( S tr i n g newColor ) { c o l o r = newColor ;  } | | |
| 25 |  | |  |
| 26 |  | |  |
| 27 | } | |  |
|  |  | |  |
| 1 | package s e r v e r ; | |  |
| 2 |  | |  |
| 3 | p u b l i c c l a s s SlowPowUp extends | | PowUp{ |
| 4 | p u b l i c SlowPowUp ( ) { | |  |
| 5 | setName ( " Slow " ) ; | |  |
| 6 | s e t C o l o r ( " #0728ab " ) ; | |  |
| 7 | } | |  |
| 8 | } | |  |
|  |  | |  |
| 1 | package s e r v e r ; | |  |
| 2 |  | |  |
| 3 | p u b l i c c l a s s DashPowUp extends | | PowUp{ |
| 4 |  | |  |
| 5 | p u b l i c DashPowUp ( ) { | |  |
| 6 | setName ( " Dash " ) ; | |  |
| 7 | s e t C o l o r ( "#ccbb23 " ) ; | |  |
| 8 | } | |  |
| 9 | } | |  |

|  |  |  |
| --- | --- | --- |
| 1  2  3  4 | package s e r v e r ;  p u b l i c c l a s s JumpPowUp extends PowUp{ p u b l i c JumpPowUp() { | |
| 5 | setName ( " Jump" ) ; |  |
| 6 | s e t C o l o r ( "#ab4907 " ) ; |  |
| 7 | } |  |
| 8 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c c l a s s TeleportPowUp extends | PowUp{ |
| 4 |  |  |
| 5 | p u b l i c TeleportPowUp ( ) { |  |
| 6 | setName ( " Teleport " ) ; |  |
| 7 | s e t C o l o r ( "#7b219e " ) ; |  |
| 8 | } |  |
| 9 | } |  |

## Builder



### pav. Builder diagrama

Builder šablono pereikis atsirado norint palengvinti kiekvieno žaidimo lygio kurimą.

package s e r v e r ;

import java . u t i l . L i s t ;

p u b l i c c l a s s Stage implements IStage Plan {

p r i v a te List <Coordinates> grounCor ; p r i v a te List <Coordinates> wall Cor ;

@Override

1

2

3

4

5

6

7

8

9

10

11

12 p u b l i c void setGrounds ( List <Coordinates> groundCor ) {

13 t h i s . grounCor = groundCor ;

14 }

15

16 @Override

17 p u b l i c void set Walls ( List <Coordinates> wall Cor ) {

18 t h i s . wall Cor = wall Cor ;

19 }

20

21 p u b l i c List <Coordinates> getGrounds ( ) {

22 return t h i s . grounCor ;

23 }

24

25 p u b l i c List <Coordinates> get Walls ( ) {

26 return t h i s . wall Cor ;

27 }

28

29

30

31

32

33

34

35

36 }

1

package s e r v e r ;

p u b l i c c l a s s Sta g e D i r e c to r {

2

3

4

p r i v a te ISta g e Bu i l d e r s ta g e Bu i l d e r ;

p u b l i c Sta g e D i r e c to r ( ISta g e Bu i l d e r s ta g e Bu i l d e r ) { t h i s . s ta g e Bu i l d e r = s ta g e Bu i l d e r ;

}

p u b l i c Stage get Stage ( ) {

return t h i s . s ta g e Bu i l d e r . get Stage ( ) ;

}

p u b l i c void makeStage ( ) {

t h i s . s ta g e Bu i l d e r . build Wall ( ) ; t h i s . s ta g e Bu i l d e r . buildGround ( ) ;

}

}

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

1

package s e r v e r ;

import java . u t i l . Array List ; import java . u t i l . L i s t ;

p u b l i c c l a s s Stage 1 Builder implements ISta g e Bu i l d e r { p r i v a te Stage s ta g e ;

p r i v a te i n t g r i d S i z e ;

p u b l i c Stage 1 Builder ( i n t g r i d S i z e ) { t h i s . s ta g e = new Stage ( ) ;

t h i s . g r i d S i z e = g r i d S i z e ;

2

3

4

5

6

7

8

9

10

11

12

13 }

14

15 @Override

16 p u b l i c void buildGround ( ) {

17 List <Coordinates> ground = new Array List<Coordinates >() ;

18

19

20

21

- 1 )

22

23

24

f o r ( i n t x = 0 ; x<g r i d S i z e ; x++){

f o r ( i n t y = 0 ; y<g r i d S i z e ; y++){

i f ( x != 0 && x != g r i d S i z e - 1 && y != 0 && y != g r i d S i z e

ground . add ( new Coordinates ( x , y ) ) ;

}

}

25

26

27

28 }

s ta g e . setGrounds ( ground ) ;

29

30 @Override

31 p u b l i c void build Wall ( ) {

32 List <Coordinates> w a l l s = new Array List<Coordinates >() ;

33

34

35

36

- 1 )

37

38

39

f o r ( i n t x = 0 ; x<g r i d S i z e ; x++){

f o r ( i n t y = 0 ; y<g r i d S i z e ; y++){

i f ( x == 0 | | x == g r i d S i z e - 1 | | y == 0 | | y == g r i d S i z e

w a l l s . add ( new Coordinates ( x , y ) ) ;

}

}

40

41 System . out . p r i n t l n ( " Koks dydis Pr i e s : " + w a l l s . s i z e ( ) ) ;

42

43 w a l l s . add ( new Coordinates ( 5 , 4 ) ) ;

44 w a l l s . add ( new Coordinates ( 5 , 6 ) ) ;

45 w a l l s . add ( new Coordinates ( 6 , 5 ) ) ;

46

47 w a l l s . add ( new Coordinates ( 15 , 16 ) ) ;

48 w a l l s . add ( new Coordinates ( 15 , 14 ) ) ;

49 w a l l s . add ( new Coordinates ( 14 , 15 ) ) ;

50

51 f o r ( i n t x = 1 ; x < 1 9 ; x++){

52 w a l l s . add ( new Coordinates ( x , 11 ) ) ;

53 }

54

55 System . out . p r i n t l n ( " Koks dydis Po : " + w a l l s . s i z e ( ) ) ;

56

57

58 }

s ta g e . set Walls ( w a l l s ) ;

59

60 p u b l i c Stage get Stage ( ) {

61 return t h i s . s ta g e ;

62 }

63

64 }

1

package s e r v e r ;

import java . u t i l . Array List ;

2

3

4 import java . u t i l . L i s t ;

5

6 p u b l i c c l a s s Stage 2 Builder implements ISta g e Bu i l d e r {

7 p r i v a te Stage s ta g e ;

8 p r i v a te i n t g r i d S i z e ;

9

10 p u b l i c Stage 2 Builder ( i n t g r i d S i z e ) {

11 t h i s . s ta g e = new Stage ( ) ;

12 t h i s . g r i d S i z e = g r i d S i z e ;

13 }

14

15 @Override

16 p u b l i c void buildGround ( ) {

17 List <Coordinates> ground = new Array List<Coordinates >() ;

18

19

20

21

- 1 )

22

23

24

f o r ( i n t x = 0 ; x<g r i d S i z e ; x++){

f o r ( i n t y = 0 ; y<g r i d S i z e ; y++){

i f ( x != 0 && x != g r i d S i z e - 1 && y != 0 && y != g r i d S i z e

ground . add ( new Coordinates ( x , y ) ) ;

}

}

25

26

27

28 }

s ta g e . setGrounds ( ground ) ;

29

30 @Override

31 p u b l i c void build Wall ( ) {

32 List <Coordinates> w a l l s = new Array List<Coordinates >() ;

33

34

35

36

- 1 )

37

38

39

f o r ( i n t x = 0 ; x<g r i d S i z e ; x++){

f o r ( i n t y = 0 ; y<g r i d S i z e ; y++){

i f ( x == 0 | | x == g r i d S i z e - 1 | | y == 0 | | y == g r i d S i z e

w a l l s . add ( new Coordinates ( x , y ) ) ;

}

}

40

41 System . out . p r i n t l n ( " Koks dydis Pr i e s : " + w a l l s . s i z e ( ) ) ;

42

43

44

45

46 f o r ( i n t i = 7 ; i <11; i++){

47 w a l l s . add ( new Coordinates ( i , 9 ) ) ;

48 w a l l s . add ( new Coordinates ( i , 13 ) ) ;

49 w a l l s . add ( new Coordinates ( i , 17 ) ) ;

50 }

51

52 f o r ( i n t i =10; i <17; i++){

53 i f ( i != 13 ) {

54 w a l l s . add ( new Coordinates ( 6 , i ) ) ;

55 w a l l s . add ( new Coordinates ( 11 , i ) ) ;

56 }

57 }

58

59

60

System . out . p r i n t l n ( " Koks dydis Po : " + w a l l s . s i z e ( ) ) ;

s ta g e . set Walls ( w a l l s ) ;

}

p u b l i c Stage get Stage ( ) { return t h i s . s ta g e ;

}

}

61

62

63

64

65

66

67

68

69

70

71

72

73

1

package s e r v e r ;

import java . u t i l . Array List ; import java . u t i l . L i s t ;

p u b l i c c l a s s Stage 3 Builder implements ISta g e Bu i l d e r { p r i v a te Stage s ta g e ;

p r i v a te i n t g r i d S i z e ;

p u b l i c Stage 3 Builder ( i n t g r i d S i z e ) { t h i s . s ta g e = new Stage ( ) ;

t h i s . g r i d S i z e = g r i d S i z e ;

}

2

3

4

5

6

7

8

9

10

11

12

13

14

15 @Override

16 p u b l i c void buildGround ( ) {

17 List <Coordinates> ground = new Array List<Coordinates >() ;

18

19

20

21

- 1 )

22

23

24

f o r ( i n t x = 0 ; x<g r i d S i z e ; x++){

f o r ( i n t y = 0 ; y<g r i d S i z e ; y++){

i f ( x != 0 && x != g r i d S i z e - 1 && y != 0 && y != g r i d S i z e

ground . add ( new Coordinates ( x , y ) ) ;

}

}

25

26

27

28 }

s ta g e . setGrounds ( ground ) ;

29

30 @Override

31 p u b l i c void build Wall ( ) {

32 List <Coordinates> w a l l s = new Array List<Coordinates >() ;

33

34

35

36

- 1 )

37

38

39

f o r ( i n t x = 0 ; x<g r i d S i z e ; x++){

f o r ( i n t y = 0 ; y<g r i d S i z e ; y++){

i f ( x == 0 | | x == g r i d S i z e - 1 | | y == 0 | | y == g r i d S i z e

w a l l s . add ( new Coordinates ( x , y ) ) ;

}

}

40

41 System . out . p r i n t l n ( " Koks dydis Pr i e s : " + w a l l s . s i z e ( ) ) ;

|  |  |  |  |
| --- | --- | --- | --- |
| 42 |  | | |
| 43 |
| 44 | w a l l s . add ( new | Coordinates ( 3 , | 14 - 1 ) ) ; |
| 45 | w a l l s . add ( new | Coordinates ( 4 , | 14 - 1 ) ) ; |
| 46 | w a l l s . add ( new | Coordinates ( 5 , | 14 - 1 ) ) ; |
| 47 | w a l l s . add ( new | Coordinates ( 6 , | 14 - 1 ) ) ; |
| 48 | w a l l s . add ( new | Coordinates ( 3 , | 18 - 1 ) ) ; |
| 49 | w a l l s . add ( new | Coordinates ( 4 , | 18 - 1 ) ) ; |
| 50 | w a l l s . add ( new | Coordinates ( 5 , | 18 - 1 ) ) ; |
| 51 | w a l l s . add ( new | Coordinates ( 6 , | 18 - 1 ) ) ; |
| 52 | w a l l s . add ( new | Coordinates ( 3 , | 17 - 1 ) ) ; |
| 53 | w a l l s . add ( new | Coordinates ( 3 , | 16 - 1 ) ) ; |
| 54 | w a l l s . add ( new | Coordinates ( 3 , | 15 - 1 ) ) ; |
| 55 | w a l l s . add ( new | Coordinates ( 5 , | 16 - 1 ) ) ; |
| 56 | w a l l s . add ( new | Coordinates ( 6 , | 16 - 1 ) ) ; |
| 57 | w a l l s . add ( new | Coordinates ( 6 , | 15 - 1 ) ) ; |
| 58 |  |  |  |
| 59 |  |  |  |
| 60 |  |  |  |
| 61 |  |  |  |
| 62 |  |  |  |
| 63 | w a l l s . add ( new | Coordinates ( 8 , | 14 - 1 ) ) ; |
| 64 | w a l l s . add ( new | Coordinates ( 8 , | 15 - 1 ) ) ; |
| 65 | w a l l s . add ( new | Coordinates ( 8 , | 16 - 1 ) ) ; |
| 66 | w a l l s . add ( new | Coordinates ( 8 , | 17 - 1 ) ) ; |
| 67 | w a l l s . add ( new | Coordinates ( 11 , | 14 - 1 ) ) ; |
| 68 | w a l l s . add ( new | Coordinates ( 11 , | 15 - 1 ) ) ; |
| 69 | w a l l s . add ( new | Coordinates ( 11 , | 16 - 1 ) ) ; |
| 70 | w a l l s . add ( new | Coordinates ( 11 , | 17 - 1 ) ) ; |

71 w a l l s . add ( new Coordinates ( 9 , 16 - 1 ) ) ;

72 w a l l s . add ( new Coordinates ( 10 , 16 - 1 ) ) ;

73 w a l l s . add ( new Coordinates ( 9 , 18 - 1 ) ) ;

74 w a l l s . add ( new Coordinates ( 10 , 18 - 1 ) ) ;

75

76

77

78 w a l l s . add ( new Coordinates ( 13 , 14 - 1 ) ) ;

79 w a l l s . add ( new Coordinates ( 14 , 14 - 1 ) ) ;

80 w a l l s . add ( new Coordinates ( 15 , 14 - 1 ) ) ;

81 w a l l s . add ( new Coordinates ( 16 , 14 - 1 ) ) ;

82

83 w a l l s . add ( new Coordinates ( 13 , 15 - 1 ) ) ;

84 w a l l s . add ( new Coordinates ( 13 , 16 - 1 ) ) ;

85 w a l l s . add ( new Coordinates ( 13 , 17 - 1 ) ) ;

86 w a l l s . add ( new Coordinates ( 13 , 18 - 1 ) ) ;

87

88

89 f o r ( i n t i = 8 ; i <12; i++){

90 w a l l s . add ( new Coordinates ( i , 11 - 1 ) ) ;

91 w a l l s . add ( new Coordinates ( i , 8 - 1 ) ) ;

92 w a l l s . add ( new Coordinates ( i , 5 - 1 ) ) ;

93 }

94

95 w a l l s . add ( new Coordinates ( 8 , 10 - 1 ) ) ;

96 w a l l s . add ( new Coordinates ( 8 , 9 - 1 ) ) ;

97 w a l l s . add ( new Coordinates ( 11 , 10 - 1 ) ) ;

98 w a l l s . add ( new Coordinates ( 11 , 9 - 1 ) ) ;

99 w a l l s . add ( new Coordinates ( 11 , 7 - 1 ) ) ;

100

w a l l s . add ( new Coordinates ( 11 , 6 - 1 ) ) ;

System . out . p r i n t l n ( " Koks dydis Po : " + w a l l s . s i z e ( ) ) ;

s ta g e . set Walls ( w a l l s ) ;

}

p u b l i c Stage get Stage ( ) { return t h i s . s ta g e ;

}

101

102

103

104

105

106

107

108

109

110

|  |  |  |
| --- | --- | --- |
| 111  112 |  |  |
| 113 | } |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c i n t e r f a c e ISta g e Bu i l d e r | { |
| 4 | p u b l i c void buildGround ( ) ; |  |
| 5 | p u b l i c void build Wall ( ) ; |  |
| 6 |  |  |
| 7 | p u b l i c Stage get Stage ( ) ; |  |
| 8 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | import java . u t i l . L i s t ; |  |
| 4 |  |  |
| 5 | p u b l i c i n t e r f a c e IStage Plan { |  |
| 6 |  |  |

7

p u b l i c void setGrounds ( List <Coordinates> groundCor ) ; p u b l i c void set Walls ( List <Coordinates> wall Cor ) ;

}

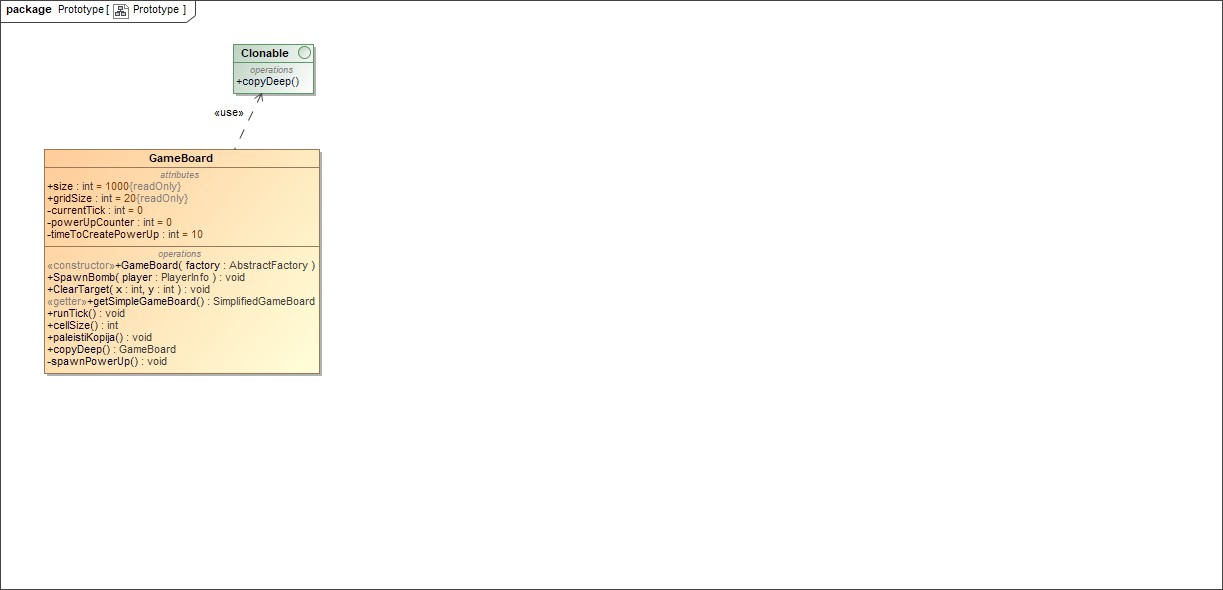
8

9

10

11

## Prototype



### pav. Prototype diagrama

Prototype šablono poreikis atsirado norint kad butu galymi gryžtai jau ankščiau išsaugotą žaidimo stadiją, kad būtų galimą į ją grižti.

# PROJEKTO APRAŠYMAS ANTRA DALIS

Žaidimas, kuriame strategiškai dėlioji bombas, kuriomis gali sunaikinti kliūtis bei priešus, išvengi- nėji spąstų bei naudoji įgytas galias kad įgautum pranašumą prieš savo varžovą.

Text

Description automatically generated

### pav. Žaidimo prototipo nuotrauka

# ŽAIDIMO REIKALAVIMAI

# ŽAIDIMO LYGIAI

Žaidimas susidės iš trijų lygių, kurie vienas nuo kito skirsis savo žaidimo strategijomis bei sudėtin- gumu.

# PIRMASIS LYGIS

Jame bus sukurtas pasaulis, kuris susidės iš sienų, kurių kiaurai pereiti negalima, bet galima jas susprogdinti naudojant bombas, bei žemės, per kurią žaidėjas gales laisvai vaikščioti. Žaidimo tiklsas nugalėti savo priešininką, taktiškai naikinant sienas. Žaidėjai turės po 3 gyvybes, viena gyvybė yra pra- randama jeigu savo arba priešininko bomba sprogsta šalia.

# ANTRAS LYGIS

Antrame lygyje atsiranda nauja kliūtis- tai spąstai, kurie sugeneruojami sukuriant 2 lygio pasaulį. Visi spąstai atrodo taip pat tik žaidėjai nežino, ką jis gali padaryti, tik žino tą kad visi spąstai jį užsaldys. Taip pat spąstai gali žaidėja sulėtinti, nuimti gyvybę ar jį kažkur nukelti.

# TREČIAS LYGIS

Trečiame lygyje atsiranda žaidėjo įgudis. Kai 3 lygio pasaulis yra sugeneruotas kas 10 sekundžių atsiranda žaidėjo įgudis, ant kurio užlipus žaidėjas gauna viena iš 4 pagerinimų:

* + Sulėtinimas priešininko. Galima naudoti kas 30 sekundžių.
  + Nusikelimas į naują vietą. Galima naudoti kas 20 sekundžių.
  + Sienų peršokimas. Galima naudoti kas 15 sekundžių.
  + Greitas žaidėjo paslinkimas. Galima naudoti kas 15 sekundžių.

# PROJEKTUI NAUDOTOS TECHNOLOGIJOS

Serverio ir kliento komunikacijai naudotas Kryonet. Programos lango sukūrimui ir piešimui buvo pasitelkta lwjgl ir OpenGL 1.1 įrankiai.

# USE CASE DIAGRAMA

Graphical user interface, diagram

Description automatically generated

### 4.2 pav. Use case diagrama

# KLASIŲ DIAGRAMA

Diagram

Description automatically generated

### pav. Klasių diagrama

# 11 ŠABLONAI

### GameState

### GameState šablono prireikė norint įdėti į žaidimą skirtingas būsenas. Tai yra :

### Žaidimo laukimas

### Žaidimo vykdymas

### Žaidimo užbaigimo bandymas

### Žaidimo lentos pakeitimas į kitą

### Šis šablonas puikiai padėjo įgyvendinti šią idėją.

### UML diagrama:

### Diagram Description automatically generated

### Kodas:

|  |  |  |  |
| --- | --- | --- | --- |
| 1  2  3  4  5  6  7  8  9 | package s e r v e r ;  p u b l i c a b s tr a c t c l a s s GameState  {  p r i v a te GameState next State ;  p r o te c te d boolean is Ready For NextStage = f a l s e ;  p u b l i c void set Next State ( GameState next State ) { t h i s . next State = next State ; | | |
| 10 | } |  |  |
| 11 |  |  |  |
| 12 | p u b l i c void get Next State ( GameServer | context | ) |
| 13 | { |  |  |
| 14 | context . s e t S ta te ( next State ) ; |  |  |
| 15 | } |  |  |
| 16 |  |  |  |
| 17 | p u b l i c a b s tr a c t void handleUpdate ( ) ; |  |  |
| 18 |  |  |  |
| 19 | p u b l i c boolean is Ready For NextStage ( ) |  |  |
| 20 | { |  |  |
| 21 | return is Ready For NextStage ; |  |  |
| 22 | } |  |  |
| 23 |  |  |  |
| 24 | } |  |  |
| 25 |  |  |  |
|  |  |  |  |
| 1 | package s e r v e r ; |  |  |
| 2 |  |  |  |

3 p u b l i c c l a s s StatePlayGame extends GameState

4 {

5 p r i v a te f i n a l i n t gameTime = 6 0 ; // 1 minute

6 p r i v a te i n t timer = 0 ;

7 p r i v a te boolean players Dead = f a l s e ;

8

9 @Override

10 p u b l i c void handleUpdate ( )

11 {

12 GameServer game Server = GameServer . g e t I n s ta n c e ( ) ;

13 t h i s . is Ready For NextStage = f a l s e ;

14 i f ( timer > gameTime \* 60 | | players Dead )

15 {

16 t h i s . is Ready For NextStage = true ;

17 t h i s . timer = 0 ;

18 t h i s . players Dead = f a l s e ;

19 }

20 e l s e

21 {

22 i f ( game Server . check I f Player Dead ( ) )

23 {

24 t h i s . players Dead = true ;

25 game Server . enableMovement = f a l s e ;

26 }

27 game Server . enableMovement = true ;

28 t h i s . timer ++;

29 }

30

31 }

32

}

33

1 package s e r v e r ;

2

3 p u b l i c c l a s s StateChangingMap extends GameState

4 {

5 p r i v a te i n t timer = - 1 ;

6

7 @Override

8 p u b l i c void handleUpdate ( )

9 {

10 GameServer game Server = GameServer . g e t I n s ta n c e ( ) ;

11 game Server . enableMovement = f a l s e ;

12 t h i s . is Ready For NextStage = f a l s e ;

13 i f ( t h i s . timer == - 1 )

14 {

15 t h i s . timer = 300 ;

16 }

17 game Server . network . send String ( " Winner p l a y e r : "+ game Server . getWinner ( ) + " map r e f r e s h in . . . " + ( t h i s . timer / 60 + 1 ) ) ;

18 i f ( timer == 0 )

19 {

20 game Server . network . send String ( n u l l ) ;

21 t h i s . is Ready For NextStage = true ;

22 t h i s . timer = - 1 ;

23 game Server . setGameLevel ( ) ;

24 }

25 e l s e

|  |  |  |
| --- | --- | --- |
| 26 | { |  |
| 27 | t h i s . timer - - ; |
| 28 | } |
| 29 |  |
| 30 | } |
| 31 |  |
| 32 | } |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c c l a s s State Wait For Players | extends GameState |
| 4 | { |  |
| 5 | p r i v a te i n t timer = - 1 ; |  |
| 6 |  |  |
| 7 | @Override |  |
| 8 | p u b l i c void handleUpdate ( ) |  |
| 9 | { |  |
| 10  11  12  13  14  15  16  17  18  19  20 | GameServer game Server = GameServer . g e t I n s ta n c e ( ) ; t h i s . is Ready For NextStage = f a l s e ;  i f ( game Server . getConnected Player Count ( ) > 2 )  {  i f ( t h i s . timer < 0 )  {  t h i s . timer = 300 ;  }  game Server . network . send String ( "Game i s s t a r t i n g in . . " + ( t h i s . timer / 60 + 1 ) ) ;  i f ( timer == 0 )  { | |

21

game Server . network . send String ( n u l l ) ; t h i s . is Ready For NextStage = true ;

t h i s . timer = - 1 ;

}

e l s e

{

t h i s . timer - - ;

}

}

e l s e

{

t h i s . timer = - 1 ;

game Server . network . send String ( " Waiting f o r p l a y e r s . . " ) ;

}

}

}

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

1

package s e r v e r ;

import java . u t i l . Array List ; import java . u t i l . L i s t ;

import shared . Vector 2 f ;

p u b l i c c l a s s StateEndingGame extends GameState

{

p r i v a te i n t timer = 0 ;

2

3

4

5

6

7

8

9

10

11 p r i v a te boolean players Dead = f a l s e ;

12 p r i v a te i n t row = 1 ;

13 p r i v a te i n t column = 1 ;

14 p r i v a te i n t row2 = 1 ;

15 p r i v a te i n t column2 = 1 ;

16 p r i v a te i n t c e n te r = 0 ;

17 @Override

18 p u b l i c void handleUpdate ( )

19 {

20 GameServer game Server = GameServer . g e t I n s ta n c e ( ) ;

21 t h i s . is Ready For NextStage = f a l s e ;

22 i f ( players Dead )

23 {

24 t h i s . is Ready For NextStage = true ;

25 t h i s . timer = 0 ;

26 t h i s . players Dead = f a l s e ;

27 row = 0 ;

28 column = 1 ;

29 row2 = 1 ;

30 column2 = 1 ;

31 c e n te r = 0 ;

32 }

33 e l s e

34 {

35 timer ++;

36 i f ( game Server . check I f Player Dead ( ) )

37 {

38 t h i s . players Dead = true ;

39 game Server . enableMovement = f a l s e ;

40

41

42

43 }

}

game Server . enableMovement = true ; place Deadly Bricks ( ) ;

44

45 }

46

47 p r i v a te void place Deadly Bricks ( )

48 {

49 GameServer game Server = GameServer . g e t I n s ta n c e ( ) ;

50 i f ( timer % 15 == 0 )

51 {

52 i f ( row < game Server . gameBoard . g r i d S i z e - 2 - c e n te r )

53 {

54 game Server . gameBoard . addObject ( row , column , " wall " ) ;

55 t h i s . damage Players ( new Vector 2 f ( row , column ) ) ;

56 row ++;

57 }

58 e l s e i f ( column < game Server . gameBoard . g r i d S i z e - 1 - c e n te r )

59 {

60 game Server . gameBoard . addObject ( row , column , " wall " ) ;

61 t h i s . damage Players ( new Vector 2 f ( row , column ) ) ;

62 column ++;

63 }

64 e l s e i f ( row2 < game Server . gameBoard . g r i d S i z e - 2 - c e n te r )

65 {

66 game Server . gameBoard . addObject ( game Server . gameBoard . g r i d S i z e - row2 - 2 , column - 1 , " wall " ) ;

67 t h i s . damage Players ( new Vector 2 f ( game Server . gameBoard . g r i d S i z e -

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 68 | row2 - 2 , column - 1 ) ) ;  row2 ++; | | | |  |
| 69 | } | | | |
| 70 | e l s e i f ( column2 < game Server . gameBoard . g r i d S i z e - 4 - c e n te r ) | | | |
| 71 | { | | | |
| 72  73 | game Server . gameBoard . addObject ( game Server . gameBoard . g r i d S i z e - row2 - 1 , game Server . gameBoard . g r i d S i z e - column2 - 2 , " wall " ) ;  t h i s . damage Players ( new Vector 2 f ( game Server . gameBoard . g r i d S i z e | | | | - |
| 74 | row2 - 1 , game Server . gameBoard . g r i d S i z e  column2 ++; | | - column2 | - 2 ) ) ; | |
| 75 | } | |  |  | |
| 76 | e l s e | |  |  | |
| 77 | { | |  |  | |
| 78 | c e n te r ++; | |  |  | |
| 79 | row = c e n te r ; | |  |  | |
| 80 | column = c e n te r + 1 ; | |  |  | |
| 81 | row2 = c e n te r + 1 ; | |  |  | |
| 82 | column2 = c e n te r + 1 ; | |  |  | |
| 83 |  | } | | | |
| 84 |  | } | | | |
| 85 | } |  | | | |
| 86 |  |  | | | |
| 87 |  | p r i v a te void damage Players ( Vector 2 f bLoc ) { | | | |
| 88 |  |  | | | |

89

GameServer s e s s i o n = GameServer . g e t I n s ta n c e ( ) ;

i n t c e l l S i z e = s e s s i o n . gameBoard . c e l l S i z e ( ) ;

System . out . p r i n t l n ( " K r e i p i u o s i " ) ;

90

91

92

93

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 94  95  96  97  98  99  100  101  102  103  104  105 | i n t kk = 0 ;  f o r ( MPPlayer p : s e s s i o n . p l a y e r s . va l u e s ( ) )  {  i f ( p . c o o r d i n a te != n u l l ) { i f ( kk != 0 ) {  List <Vector 2 f> pLoc = new Array List <>() ; | | | | | |
| 106 |  | i n t | x1 | = | ( i n t ) p . c o o r d i n a te . x / c e l l S i z e ; |  |
| 107 |  | i n t | y1 | = | ( i n t ) p . c o o r d i n a te . y / c e l l S i z e ; |  |
| 108 |  |  |  |  |  |  |
| 109  110 | ; | i n t  i n t | x2  y2 | =  = | ( i n t ) ( p . c o o r d i n a te . x + p . s i z e ) /  ( i n t ) p . c o o r d i n a te . y / c e l l S i z e ; | c e l l S i z e |
| 111 |  |  |  |  |  |  |
| 112 |  | i n t | x3 | = | ( i n t ) p . c o o r d i n a te . x / c e l l S i z e ; |  |
| 113 | ; | i n t | y3 | = | ( i n t ) ( p . c o o r d i n a te . y + p . s i z e ) / | c e l l S i z e |
| 114 |  |  |  |  |  |  |
| 115  116 | ; | i n t  i n t | x4  y4 | =  = | ( i n t ) ( p . c o o r d i n a te . x + p . s i z e ) /  ( i n t ) ( p . c o o r d i n a te . y + p . s i z e ) / | c e l l S i z e  c e l l S i z e |
|  | ; |  |  |  |  |  |
| 117 |  |  |  |  |  |  |
| 118 | Vector 2 f xy1 = new Vector 2 f ( x1 , y1 ) ; | | | | | |

( xy3 ) ) {

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 119 | Vector 2 f | xy2 | = | new | Vector 2 f ( x2 , | y2 ) ; |
| 120 | Vector 2 f | xy3 | = | new | Vector 2 f ( x3 , | y3 ) ; |
| 121 | Vector 2 f | xy4 | = | new | Vector 2 f ( x4 , | y4 ) ; |
| 122 |  |  |  |  |  |  |
| 123 |  |  |  |  |  |  |
| 124  125  126  127  128  129  130  131  132  133  134  135  136  137 | pLoc . add ( xy1 ) ;  i f ( ! xy1 . i s Equal ( xy2 ) ) { pLoc . add ( xy2 ) ;  System . out . p r i n t l n ( " Pridedu 2 " ) ;  }  i f ( ! xy1 . i s Equal ( xy3 ) && ! xy2 . i s Equal pLoc . add ( xy3 ) ;  System . out . p r i n t l n ( " Pridedu 3 " ) ;  }  i f ( ! xy1 . i s Equal ( xy4 ) && ! xy2 . i s Equal  i s Equal ( xy4 ) ) {  pLoc . add ( xy4 ) ; | | | | | |

( xy4 ) && ! xy3 .

138 System . out . p r i n t l n ( " Pridedu 4 " ) ;

139 }

140

141

142 DamgePlayer ( p , pLoc , bLoc ) ;

143

144 }

145

146 kk++;

147

148 }

149 }

150

151 }

152

153 p r i v a te void DamgePlayer ( MPPlayer p , List <Vector 2 f> pLoc , Vector 2 f bLoc ) {

154

155 System . out . p r i n t l n ( " Bombos : " + bLoc . x + " " + bLoc . y ) ;

156

157 f o r ( Vector 2 f Pxy : pLoc )

158 {

159 i f ( bLoc . i s Equal ( Pxy) )

160 {

161 System . out . p r i n t l n ( " Crushed by wall " ) ;

162 p . health = 0 ;

163 return ;

164 }

165 }

166

167 }

168

169 }

1

package s e r v e r ;

import java . i o . Buffered Reader ;

2

3

4

5 import java . i o . IOException ;

6 import java . i o . InputStream Reader ;

7 import java . u t i l . Date ;

8 import java . u t i l . HashMap ;

9 import java . u t i l . Map;

10 import java . u t i l . Timer ;

11 import java . u t i l . TimerTask ;

12

13 import shared . Vector 2 f ;

14 import shared . PacketUpdate Player Pos ;

15

16 c l a s s GameServer

17 {

18 p r i v a te s t a t i c GameServer game Server = n u l l ;

19 //

20 // t h i s . s ta g e 1 b u i l d e r = new Stage 1 Builder ( g r i d S i z e ) ;

21 // t h i s . s ta g e 2 b u i l d e r = new Stage 2 Builder ( g r i d S i z e ) ;

22 // t h i s . s ta g e 3 b u i l d e r = new Stage 3 Builder ( g r i d S i z e ) ;

23 //

24

25

26 p r o te c te d v o l a t i l e GameBoard gameBoard ;

27

28 p r o te c te d GameState c u r r e n t S ta te ;

29

30 p r o te c te d v o l a t i l e Map<I nteger , MPPlayer> p l a y e r s ;

31 p r o te c te d v o l a t i l e Network network ;

32 p r i v a te GameCycleThread thread ;

33 p r i v a te Stage 1 Factory s ta g e 1 f a c to r y ;

34

35 p r i v a te i n t gameLevel ;

36 p r i v a te i n t counter ;

37 p r i v a te boolean readyForNextFrame ;

38 p r i v a te Chain chain 1 ;

39 p r i v a te Chain chain 2 ;

40 p r i v a te Chain chain 3 ;

41 p r i v a te Chain chain 4 ;

42 p u b l i c boolean enableMovement = f a l s e ;

43

44 p u b l i c ItemComponent powerUp ;

45

46 p u b l i c ItemComponent every Item ;

47

48 p u b l i c Item dash ;

49 p u b l i c Item t e l e p o r t ;

50 p u b l i c Item jump ;

51 p u b l i c Item slowDown ;

52 p u b l i c Item past ;

53

54 p u b l i c Item bomb ;

55

56

57 p u b l i c Item wall ;

58

59

60

61 p u b l i c Item List i tems ;

62

63

64 p r i v a te i n t level Change Counter ;

65

66

67 p r i v a te boolean updateBoard = f a l s e ;

68

69

70 p r i v a te GameServer ( )

71 {

72

73 powerUp = new ItemGroup ( " Powerup " , " Folder that has a l l powerups that have been used " ) ;

74

75 every Item = new ItemGroup ( " Item l i s t " , " This i s the l i s t that hold data o f PowerUps , Walls and Bombs" ) ;

76

77

78

79 dash = new Item ( " Dash " , " Player moves f a s t . " , 0 ) ;

80 t e l e p o r t = new Item ( " Teleport " , " Player t e l e p o r t s . " , 0 ) ;

81 slowDown = new Item ( " SlowDown " , " Enemy get slowed down . " , 0 ) ;

82 jump = new Item ( " Jump" , " Player i s capable o f jumping over w a l l s . "

, 0 ) ;

83 past = new Item ( " Past " , " Player i s time t r a v e l e r . " , 0 ) ;

84

85

86 bomb = new Item ( "Bomb" , " Dangerous item . " , 0 ) ;

87 wall = new Item ( " Wall " , " Blocks p l a y e r s path . " , 0 ) ;

88

89

90 every Item . add ( powerUp ) ;

91

92 powerUp . add ( dash ) ;

93 powerUp . add ( t e l e p o r t ) ;

94 powerUp . add ( slowDown ) ;

95 powerUp . add ( jump ) ;

96 powerUp . add ( past ) ;

97

98 every Item . add (bomb) ;

99 every Item . add ( wall ) ;

100

101

102

103 i tems = new Item List ( every Item ) ;

104

105

106

107 counter = 0 ;

108

109 level Change Counter = 0 ;

110

111 // I n i t Connection

112 i f ( ! i n i t C o n n e c t i o n ( ) )

113 {

114 System . e r r . p r i n t l n ( "ERROR Connecting to host " ) ;

115 return ;

116 }

117

118 gameLevel = 2 ;

119

120 t h i s . p l a y e r s = new HashMap<Integer , MPPlayer >() ;

121

122 setGameLevel ( ) ;

123

124 t h i s . thread = new GameCycleThread ( ) ;

125 t h i s . thread . s t a r t ( ) ;

126

127 chain 1 = new Generate Walls ( ) ;

128 chain 3 = new GenerateDesWall ( ) ;

129 chain 2 = new GenerateTrap ( ) ;

130 readyForNextFrame = true ;

131 chain 1 . setNextChain ( chain 2 ) ;

132 chain 2 . setNextChain ( chain 3 ) ;

133

134 GameState wait For Players State = new State Wait For Players ( ) ;

135 GameState playGameState = new StatePlayGame ( ) ;

136 GameState endingGameState = new StateEndingGame ( ) ;

137 GameState changingMapState = new StateChangingMap ( ) ;

138

139 wait For Players State . set Next State ( playGameState ) ;

140 playGameState . set Next State ( endingGameState ) ;

141 endingGameState . set Next State ( changingMapState ) ;

142 changingMapState . set Next State ( wait For Players State ) ;

143 c u r r e n t S ta te = wait For Players State ;

144

145

146 }

147

148 p r o te c te d GameServer get Context ( )

149 {

150 return t h i s ;

151 }

152

153 p u b l i c boolean check I f Player Dead ( ) {

154 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

155 {

156 i f ( p . c o o r d i n a te != n u l l ) {

157 i f ( p . health < 1 && p . can Die ) {

158 p . death Counter +=1;

159 System . out . p r i n t l n ( " Mires " ) ;

160 // setGameLevel ( ) ;

161 return true ;

162 }

163 }

164 }

165 return f a l s e ;

166

167 }

168

169 p u b l i c S tr i n g getWinner ( ) {

170 boolean f i r s t C y c l e = true ;

171 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

172 {

173 i f ( ! f i r s t C y c l e )

174 {

175 i f ( p . c o o r d i n a te != n u l l ) {

176 i f ( p . health > 0 ) {

177 // setGameLevel ( ) ;

178 return S tr i n g . value Of ( p . c . getID ( ) ) ;

179 }

180 }

181 }

182 f i r s t C y c l e = f a l s e ;

183 }

184 return " None " ;

185

186 }

187

188 p u b l i c i n t getConnected Player Count ( ) {

189 return p l a y e r s . s i z e ( ) ;

190

191 }

192 // nustatome s e k a n c i a busena

193 p u b l i c void s e t S ta te ( GameState next State )

194 {

195 t h i s . c u r r e n t S ta te = next State ;

196 }

197

198

199 p u b l i c void setGameLevel ( ) {

200

201

202

203 i f ( gameLevel == 3 )

204 gameLevel = 1 ;

205 e l s e

206 gameLevel++;

207

208

209 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

210 {

211 i f ( p . c o o r d i n a te != n u l l ) {

212 System . out . p r i n t l n ( " Atstatau gyvyvbes " ) ;

213 p . health = p . base Health ;

214 p . speed = p . base Speed ;

215

216 }

217 }

218

219

220

221 ChangeLevel ( gameLevel ) ;

222

223 }

224 p u b l i c void s to p Pl a y e r s ( )

225 {

226 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

227 {

228 i f ( p . c o o r d i n a te != n u l l ) {

229 p . isHoldingDown = f a l s e ;

230 p . is Holding Up = f a l s e ;

231 p . i s H o l d i n g L e f t = f a l s e ;

232 p . i s Holding Right = f a l s e ;

233 p . i s H o l d i n g S k i l l = f a l s e ;

234 p . i s Holding Use = f a l s e ;

235 }

236

237

238 }

239 }

240

241 p u b l i c void ChangeLevel ( i n t l e v e l ) {

242

243 level Change Counter = 60 \* 2 ;

244

245 s to p Pl a y e r s ( ) ;

246 gameLevel = l e v e l ;

247

248 ISta g e Bu i l d e r b u i l d e r ;

249 Abstract Factory f a c to r y ;

250 switch ( l e v e l ) {

251 case 1 :

252 d e f a u l t :

253 b u i l d e r = new Stage 1 Builder ( 20 ) ;

254 f a c to r y = new Stage 1 Factory ( ) ;

255 t h i s . gameBoard = new GameBoard( f ac tory , bu i l de r , 1 ) ;

256 break ;

257 case 2 :

258 b u i l d e r = new Stage 2 Builder ( 20 ) ;

259 f a c to r y = new Stage 1 Factory ( ) ;

260 t h i s . gameBoard = new GameBoard( f ac tory , bu i l de r , 2 ) ;

261 break ;

262 case 3 :

263 b u i l d e r = new Stage 3 Builder ( 20 ) ;

264 f a c to r y = new Stage 1 Factory ( ) ;

265 t h i s . gameBoard = new GameBoard( f ac tory , bu i l de r , 3 ) ;

266 break ;

267 }

268 respawn All Players ( ) ;

269 refresh Map For All Players ( ) ;

270

271

272

273 }

274

275 p u b l i c s t a t i c GameServer g e t I n s ta n c e ( )

276 {

277 i f ( game Server == n u l l )

278 {

279 game Server = new GameServer ( ) ;

280 }

281 return game Server ;

282 }

283

284 p r i v a te boolean i n i t C o n n e c t i o n ( )

285 {

286 t h i s . network = new Network ( ) ;

287

288 i f ( ! t h i s . network . i n i t Kr y o Se r v e r ( ) )

289 {

290 return f a l s e ;

291 }

292

293 return true ;

294 }

295

296 p u b l i c void respawn All Players ( )

297 {

298 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

299 {

300 i f ( p . c o o r d i n a te != n u l l ) {

301 t h i s . respawn Player ( p ) ;

302 }

303

304

305 }

306

307 }

308

309 p u b l i c void respawn Player ( Pl a y e r In f o p )

310 {

311 GameServer gameserver = GameServer . g e t I n s ta n c e ( ) ;

312 boolean te l e p o r te d = f a l s e ;

313 i n t maxRetry = 6 0 ;

314 i n t r e t r y = 0 ;

315 while ( ! te l e p o r te d && r e t r y < maxRetry )

316 {

317 i n t randomCoordX = getRandomNumber ( 0 , gameserver . gameBoard . g r i d S i z e ) ;

318 i n t randomCoordY = getRandomNumber ( 0 , gameserver . gameBoard . g r i d S i z e ) ;

319 i f ( gameserver . gameBoard . o b j e c t s [ randomCoordX ] [ randomCoordY ] i n s t a n c e o f Ground )

320 {

321 p . c o o r d i n a te . x = randomCoordX \* ( gameserver . gameBoard . s i z e / gameserver . gameBoard . g r i d S i z e ) ;

322 p . c o o r d i n a te . y = randomCoordY \* ( gameserver . gameBoard . s i z e / gameserver . gameBoard . g r i d S i z e ) ;

323 te l e p o r te d = true ;

324 }

325 r e t r y ++;

326 }

327 }

328

329 p r i v a te i n t getRandomNumber ( i n t min , i n t max)

330 {

331 return ( i n t ) ( ( Math . random ( ) \* (max - min ) ) + min ) ;

332 }

333

334 p u b l i c void update Player ( i n t id , PacketUpdate Player Pos player Packet )

335 {

336 MPPlayer p l a y e r = p l a y e r s . get ( id ) ;

337 i f ( p l a y e r != n u l l && level Change Counter < 0 )

338 {

339 p l a y e r . is Holding Up = player Packet . is Holding Up != n u l l ? player Packet . is Holding Up : p l a y e r . is Holding Up ;

340 p l a y e r . isHoldingDown = player Packet . isHoldingDown != n u l l ? player Packet . isHoldingDown : p l a y e r . isHoldingDown ;

341 p l a y e r . i s H o l d i n g L e f t = player Packet . i s H o l d i n g L e f t != n u l l ? player Packet . i s H o l d i n g L e f t : p l a y e r . i s H o l d i n g L e f t ;

342 p l a y e r . i s Holding Right = player Packet . i s Holding Right != n u l l ? player Packet . i s Holding Right : p l a y e r . i s Holding Right ;

343 p l a y e r . i s Holding Use = player Packet . i s Holding Use != n u l l ? player Packet . i s Holding Use : p l a y e r . i s Holding Use ;

344 p l a y e r . i s H o l d i n g S k i l l = player Packet . i s H o l d i n g S k i l l != n u l l ? player Packet . i s H o l d i n g S k i l l : p l a y e r . i s H o l d i n g S k i l l ;

345 p l a y e r s . put ( p l a y e r . id , p l a y e r ) ;

346 }

347 }

348

349 p u b l i c void add Player ( MPPlayer p l a y e r )

350 {

351 t h i s . p l a y e r s . put ( p l a y e r . c . getID ( ) , p l a y e r ) ;

352 t h i s . respawn Player ( p l a y e r ) ;

353 t h i s . network . sendGameBoard ( gameBoard , p l a y e r ) ;

354

355 }

356

357 p u b l i c void refresh Map For All Players ( )

358 {

359 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

360 {

361 network . s en d Player In f o ( p , true ) ;

362 t h i s . network . sendGameBoard ( gameBoard , p ) ;

363 }

364 }

365

366 p u b l i c void remove Player ( i n t id )

367 {

368

369 p l a y e r s . remove ( id ) ;

370 }

371

372

373 p r i v a te c l a s s GameCycleThread extends Thread

374 {

375 v o l a t i l e boolean isGameRunning = true ;

376 p r i v a te f i n a l i n t gameSpeed = 1 6 ; //The lower the number the f a s t e r the game i s

377

378

379

380 p u b l i c GameCycleThread ( )

381 {

382 t h i s . isGameRunning = true ;

383 }

384

385 p u b l i c void run ( )

386 {

387 long last Time = System . nanoTime ( ) ;

388 f i n a l double ns = 1000000000 . 0 / 6 0 . 0 ;

389 double d e l ta = 0 ;

390 while ( true ) {

391 long now = System . nanoTime ( ) ;

392 d e l ta += ( now - last Time ) / ns ;

393 last Time = now ;

394 while ( d e l ta >= 1 ) {

395 update ( ) ;

396 delta - - ;

397 }

398 }

399 }

400

401 p u b l i c void stopGame ( )

402 {

403 t h i s . isGameRunning = f a l s e ;

404

405 // network should probably be c l o s e d by the parent

406 network . c l o s e ( ) ;

407 }

408

409 p r i v a te void update ( )

410 {

411

412 counter++;

413 level Change Counter - - ;

414 c u r r e n t S ta te . handleUpdate ( ) ;

415 i f ( c u r r e n t S ta te . is Ready For NextStage ( ) )

416 {

417 c u r r e n t S ta te . get Next State ( get Context ( ) ) ;

418 }

419

420 i f ( enableMovement )

421 {

422 update Players ( ) ;

423 gameBoard . run Tick ( ) ;

424 }

425 surenkamTeksta ( ) ;

426

427

428 check I f Player Dead ( ) ;

429

430

431

432

433 }

434

435

436

437 p r i v a te void surenkamTeksta ( ) {

438 InputStream Reader f i l e I n p u t S tr e a m=new InputStream Reader ( System . in

) ;

439 Buffered Reader buffered Reader=new Buffered Reader ( f i l e I n p u t S tr e a m )

;

440

441 S tr i n g te k s ta s = " nieko nenuskaityta " ;

442

443

444 try {

445 i f ( buffered Reader . ready ( ) ) {

446

447 te k s ta s = buffered Reader . read Line ( ) ;

448

449

450

451 System . out . p r i n t l n ( " i r a s y t a s te k s ta s " + te k s ta s ) ;

452

453

454

455 S tr i n g tekstas Lower = te k s ta s . toLowerCase ( ) ;

456

457

458 S tr i n g [ ] s a r a s a s = tekstas Lower . s p l i t ( " " ) ;

459

460

461 i f ( s a r a s a s . l ength == 1 ) {

462

463 i tems . g e t I te m L i s t ( ) ;

464

465 }

466 e l s e i f ( s a r a s a s . l ength > 3 ) {

467 Conversion Context task = new Conversion Context ( te k s ta s ) ;

468

469 S tr i n g whatToPlace = task . getWhat ( ) ;

470 S tr i n g whereToPlace = task . getWhere ( ) ;

471 i n t howManyToPlace = task . get Quantity ( ) ;

472

473

474 Task newTask = new Task ( howManyToPlace , whatToPlace , whereToPlace , gameBoard ) ;

475

476 gameBoard = chain 1 . r e s u l t ( newTask ) ;

477

478 System . out . p r i n t l n ( whatToPlace + " How many : " + howManyToPlace ) ;

479 } e l s e i f ( s a r a s a s . l ength > 1 ) {

480

481

482

483

484

) ) {

i f ( s a r a s a s [ 0 ] . e qu a l s ( " change " ) && s a r a s a s [ 1 ] . e q u a l s ( " l e v e l "

i n t l y g i s ; try {

l y g i s = I n te g e r . p a r s e I n t ( s a r a s a s [ 2 ] ) ;

485

486 i f ( l y g i s > 0 && l y g i s < 4 ) {

487 ChangeLevel ( l y g i s ) ;

488

489

490

491

l y g i s ) ;

} e l s e {

System . out . p r i n t l n ( " Toks l y g i s n e e g z i s tu o ja : " +

}

492

493 } catch ( NumberFormatException e ) {

494 System . out . p r i n t l n ( " Toks l y g i s n e e g z i s tu o ja : " + s a r a s a s [ 2 ] ) ;

495 }

496 }

497 e l s e {

498 System . out . p r i n t l n ( " Unknown command " + te k s ta s ) ;

499 }

500

501

502 } e l s e

503 {

504 System . out . p r i n t l n ( " Unknown command " + te k s ta s ) ;

505 }

506

507

508

509 }

510

511

512 } catch ( IOException e ) {

513 e . print Stack Trace ( ) ;

514 }

515

516

517

518

519

520

521

522

523

524 }

525

526

527 p r i v a te void update Players ( )

528 {

529 i n t k = 0 ;

530 f o r ( MPPlayer p : p l a y e r s . v a l u e s ( ) )

531 {

532 i f ( k!= 0 )

533 {

534 i f ( p . i s H o l d i n g S k i l l )

535 {

536 p . tr y U s i n g S p e l l ( ) ;

537 }

538

539 p . onTick ( ) ;

540 p . reduce Timer ( ) ;

541

542 i f ( p . i s Holding Use )

543 {

544

545 i f ( p . bombTimer < 0 ) {

546 gameBoard . SpawnBomb( p ) ;

547

548 p . setBombTimer ( 2 ) ;

549 }

550

551 }

552 p . c o o r d i n a te = c h e c k C o l l i s i o n ( p ) ;

553 network . sendGameBoard ( gameBoard , p ) ;

554 network . s en d Player In f o ( p , true ) ;

555 }

556 k++;

557 }

558 }

559

560 p r i v a te Vector 2 f c h e c k C o l l i s i o n ( MPPlayer p )

561 {

562

563 Vector 2 f coords After Move = new Vector 2 f ( p . c o o r d i n a te . x , p . c o o r d i n a te . y ) ;

564

565 f l o a t padding = 0 . 001 f ;

566 f l o a t c e l l S i z e = gameBoard . c e l l S i z e ( ) ;

567

568 boolean moveX = true ;

569 boolean moveY = true ;

570

571 i f ( p . i s H o l d i n g L e f t )

572 {

573 coords After Move . x -= p . speed ;

574 }

575

576 i f ( p . i s Holding Right )

577 {

578

579 coords After Move . x += p . speed ;

580 }

581

582 boolean c o l l i d i n g L e f t = ( ( i n t ) coords After Move . x / c e l l S i z e - padding ) < ( ( i n t ) p . c o o r d i n a te . x / c e l l S i z e ) ;

583 boolean c o l l i d i n g R i g h t = ( ( ( i n t ) coords After Move . x + p . s i z e + padding ) / c e l l S i z e ) > ( ( ( i n t ) p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

584 boolean i s C o l l i d i n g X = c o l l i d i n g L e f t | | c o l l i d i n g R i g h t ;

585 moveX = ! ( coords After Move . x <= 0 | | coords After Move . x >= gameBoard . s i z e - p . s i z e ) ;

586 //Some smoothing when going around edges would be n i c e

587 i f ( i s C o l l i d i n g X && moveX)

588 {

589 i n t x = 0 , y = 0 , y1 = 0 ;

590 i f ( c o l l i d i n g L e f t )

591 {

592 x = ( i n t ) ( ( coords After Move . x ) / c e l l S i z e ) ;

593 y = ( i n t ) ( p . c o o r d i n a te . y / c e l l S i z e ) ;

594 y1 = ( i n t ) ( ( p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

595

596 }

597 i f ( c o l l i d i n g R i g h t )

598 {

599 x = ( i n t ) ( ( coords After Move . x + p . s i z e ) / c e l l S i z e ) ;

600 y = ( i n t ) ( p . c o o r d i n a te . y / c e l l S i z e ) ;

601 y1 = ( i n t ) ( ( p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

602

603 }

604

605 i f ( y == y1 )

606 {

607 moveX = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable ;

608 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

609 }

610 e l s e

611 {

612 moveX = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable && gameBoard . o b j e c t s [ x ] [ y1 ] . i s Walkable ;

613 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

614 gameBoard . o b j e c t s [ x ] [ y1 ] . onStep ( p ) ;

615 }

616 }

617

618

619 i f ( p . is Holding Up )

620 {

621 coords After Move . y += p . speed ;

622 }

623

624 i f ( p . isHoldingDown )

625 {

626 coords After Move . y -= p . speed ;

627 }

628

629 boolean c o l l i d i n g U p = ( ( ( i n t ) coords After Move . y + p . s i z e - padding

) / c e l l S i z e ) > ( ( ( i n t ) p . c o o r d i n a te . y + p . s i z e ) / c e l l S i z e ) ;

630 boolean colliding Down = ( ( i n t ) coords After Move . y / c e l l S i z e + padding ) < ( ( i n t ) p . c o o r d i n a te . y / c e l l S i z e ) ;

631 boolean i s C o l l i d i n g Y = c o l l i d i n g U p | | colliding Down ;

632 moveY = ! ( coords After Move . y <= 0 | | coords After Move . y >= gameBoard . s i z e - p . s i z e ) ;

633

634 //Some smoothing when going around edges would be n i c e

635 i f ( i s C o l l i d i n g Y && moveY)

636 {

637 i n t x = 0 , x1 = 0 , y = 0 ;

638

639 i f ( c o l l i d i n g U p )

640 {

641 y = ( i n t ) ( ( coords After Move . y + p . s i z e ) / c e l l S i z e ) ;

642 x = ( i n t ) ( p . c o o r d i n a te . x / c e l l S i z e ) ;

643 x1 = ( i n t ) ( ( p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

644

645 }

646 i f ( colliding Down )

647 {

648 y = ( i n t ) ( ( coords After Move . y ) / c e l l S i z e ) ;

649 x = ( i n t ) ( p . c o o r d i n a te . x / c e l l S i z e ) ;

650 x1 = ( i n t ) ( ( p . c o o r d i n a te . x + p . s i z e ) / c e l l S i z e ) ;

651 }

652

653 i f ( x == x1 )

654 {

655 moveY = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable ;

656 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

657 }

658 e l s e

659 {

660 moveY = gameBoard . o b j e c t s [ x ] [ y ] . i s Walkable && gameBoard . o b j e c t s [ x1 ] [ y ] . i s Walkable ;

661 gameBoard . o b j e c t s [ x ] [ y ] . onStep ( p ) ;

662 gameBoard . o b j e c t s [ x1 ] [ y ] . onStep ( p ) ;

663 }

664 }

665

666 coords After Move . x = moveX ? coords After Move . x : p . c o o r d i n a te . x ;

667 coords After Move . y = moveY ? coords After Move . y : p . c o o r d i n a te . y ;

668

669 return coords After Move ;

670

}

}

}

671

672

673

674

675

### Proxy šablonas

Kuriant projektą prireikė efektyvaus būdo apdoroti nuotraukas, kad nereikėtų į atmintį iš naujo įkėlinėti ir naudoti tik tada kada reikia. Taip pat kilo noras tekstūras šiek tiek modifikuoti jau įkėlus į atmintį. Šiuos pageidavimus tikslingai padeda įvykdyti Proxy šablonas

**UML diagrama:**

Diagram

Description automatically generated

**Greitaveika lyginant naudojima be šio šablono:**

Text

Description automatically generated

Kaip matome skirtumas labia didelis

**Kodas**

### 

package c l i e n t ;

p u b l i c c l a s s Image Character Proxy implements Image File { Image File f i l e ;

S tr i n g path ;

boolean i s Main Character = f a l s e ; S tr i n g user = n u l l ;

i n t width ; i n t height ;

Image File parentProxy ;

p u b l i c Image Character Proxy ( boolean is Main Character , S tr i n g newPath , i n t width , i n t height )

{

t h i s . i s Main Character = i s Main Character ; t h i s . path = newPath ;

t h i s . width = width ;

t h i s . height = height ;

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21 }

22

23 p u b l i c Image Character Proxy ( Image File another Proxy , boolean i s Main Character ) {

24 parentProxy = another Proxy ;

25 t h i s . i s Main Character = i s Main Character ;

26 }

27

28

29 @Override

30 p u b l i c void drawImage ( i n t x , i n t y , f l o a t r , f l o a t g , f l o a t b )

31 {

32 i f ( parentProxy == n u l l )

33 {

34 i f ( f i l e == n u l l )

35 {

36 f i l e = new Heavy Image File ( t h i s . path , t h i s . width , t h i s . height ) ;

37 t h i s . draw ( x , y , y , g , b , f i l e ) ;

38 }

39 e l s e

40 {

41 t h i s . draw ( x , y , y , g , b , f i l e ) ;

42 }

43

44 }

45 e l s e

46 {

47 t h i s . draw ( x , y , y , g , b , parentProxy ) ;

48 }

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 49 | } |  | | | | | | |
| 50 |  |
| 51 | p r i v a te void draw ( i n t x , i n t y , i n t | r , | f l o a t | g , | f l o a t | b , | Image File | f ) |
| 52 | { |  | | | | | | |
| 53 |  |
| 54 | i f ( i s Main Character ) |
| 55 | { |
| 56 | // Removing c o l o r f o r main char |
| 57 | f . drawImage ( x , y , 1 f , 1 f , 1 f ) ; |
| 58 | } |
| 59 | e l s e |
| 60 | { |
| 61 | f . drawImage ( x , y , r , g , b ) ; |

62

f l o a t s i z e = 4 0 ;

f l o a t xEn = x\*4/ s i z e ; f l o a t yEn = y\*4/ s i z e ;

Text . draw String ( " enemy " , xEn - 8/10 , yEn - 1 , s i z e , 2 ) ;

}

}

}

63

64

65

66

67

68

1

package c l i e n t ;

import java . awt . image . Buffered Image ; import java . i o . F i l e ;

import java . i o . IOException ;

import javax . imageio . ImageIO ;

2

3

4

5

6

7

8

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 9 |  | | | | | |
| 10 | p u b l i c c l a s s Heavy Image File implements Image File { | | | | |  |
| 11 |  | | | | |  |
| 12 | S p r i te heavy Sprite ; | | | | |  |
| 13 | S tr i n g path ; | | | | |  |
| 14 | p r i v a te Texture Loader texture Loader ; | | | | |  |
| 15 |  | | | | |  |
| 16 | p u b l i c Heavy Image File ( S tr i n g newPath , i n t width , i n t height ) | | | | |  |
| 17 | { | | | | |  |
| 18 | texture Loader = new Texture Loader ( ) ; | | | | |  |
| 19  20 | t h i s . heavy Sprite = new S p r i te ( texture Loader , newPath , width ,  ) ;  } | | | | | height |
| 21 |  | | | | |  |
| 22 | @Override |  |  |  |  |  |
| 23 | p u b l i c void drawImage ( i n t x , i n t | y , f l o a t r , f l o a t | g , | f l o a t | b ) | { |
| 24 | t h i s . heavy Sprite . draw ( x , y , r , | g , b ) ; |  |  |  |  |
| 25 |  |  |  |  |  |  |
| 26 | } |  |  |  |  |  |
| 27 |  |  |  |  |  |  |
| 28 | } |  |  |  |  |  |
|  |  |  |  |  |  |  |
| 1 | package c l i e n t ; |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 | p u b l i c c l a s s ImageProxy implements | Image File { |  |  |  |  |
| 4 |  |  |  |  |  |  |
| 5  6  7 | Heavy Image File f i l e = n u l l ; // heavy image f i l e i n t width ;  i n t height ; | | | | | |

8 // Heavy Image File f i l e ;

9

10 S tr i n g path ;

11 Image File parentProxy ;

12

13 p u b l i c ImageProxy ( S tr i n g newPath , i n t width , i n t height )

14 {

15 t h i s . path = newPath ;

16 t h i s . width = width ;

17 t h i s . height = height ;

18 }

19

20 p u b l i c ImageProxy ( Image File another Proxy ) {

21 parentProxy = another Proxy ;

22 }

23

24 @Override

25 p u b l i c void drawImage ( i n t x , i n t y , f l o a t r , f l o a t g , f l o a t b )

26 {

27 i f ( parentProxy == n u l l )

28 {

29 i f ( f i l e == n u l l )

30 {

31 f i l e = new Heavy Image File ( path , width , height ) ;

32 }

33 f i l e . drawImage ( x , y , r , g , b ) ;

34 }

35 e l s e

36 {

37

parentProxy . drawImage ( x , y , r , g , b ) ;

}

}

}

38

39

40

41

42

43

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | package c l i e n t ; |  | | | | | | | | |
| 2 |  |
| 3 |  |
| 4 | p u b l i c i n t e r f a c e Image File |
| 5 | { |
| 6 |  |
| 7 | p u b l i c void drawImage ( i n t | x , | i n t | y , | f l o a t | r , | f l o a t | g , | f l o a t | b ) ; |
| 8 |  |  |  |  |  |  |  |  |  |  |
| 9 | } |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| 1 | package c l i e n t ; |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |  |
| 3  4  5  6  7  8  9  10  11  12 | import java . nio . Byte Buffer ; import java . u t i l . HashMap ; import java . u t i l . Map; import org . l w j g l . glfw .GLFW;  import org . l w j g l . opengl . GL11 ;  import shared . Simplified Game Board ; import shared . S i m p l i f i e d P l a y e r ; import shared . Vector 2 f ;  import shared . ObjectType ; | | | | | | | | | |

13

14 import org . l w j g l . opengl .GL;

15

16

17

18 p u b l i c c l a s s GameWindow implements UpdateGameDataDelegate

19 {

20 p r i v a te long window ;

21 f i n a l p r i v a te i n t SCREEN\_LENGTH = 1000 ;

22 f i n a l p r i v a te i n t SCREEN\_WIDTH = 1000 ;

23 p r i v a te Simplified Game Board board ;

24 p r i v a te Network network ;

25 p r i v a te S i m p l i f i e d P l a y e r main Player ;

26 p r i v a te Map<I nteger , Client Player > p l a y e r s ;

27 p r i v a te S tr i n g d i s p l a y S t r i n g = n u l l ;

28 // Textures

29 p r i v a te Image File bombSprite ;

30 p r i v a te Image File w a l l S p r i te ;

31 p r i v a te Image File ground Sprite ;

32 p r i v a te Image File powerup Sprite ;

33 p r i v a te Image File t r a p S p r i te ;

34 p r i v a te Image File f i r e S p r i t e ;

35 p r i v a te Image File p l a y e r S p r i te ;

36

37 p u b l i c GameWindow()

38 {

39

40 // I n i t Window

41 i f ( ! initWindow ( ) )

42 {

43

44

45 }

System . e r r . p r i n t l n ( "ERROR WHILE INITIATING WINDOW" ) ; return ;

46

47 // I n i t Connection

48 i f ( ! i n i t C o n n e c t i o n ( ) )

49 {

50 System . e r r . p r i n t l n ( "ERROR Connecting to host " ) ;

51 return ;

52 }

53

54

55

;

56

57

58 }

t h i s . main Player = new S i m p l i f i e d P l a y e r ( t h i s . network . c l i e n t . getID ( ) )

t h i s . p l a y e r s = new HashMap<Integer , Client Player >() ; runGame ( ) ;

59

60

61 p r i v a te void i n i t P r o x i e s ( i n t s i z e )

62 {

63 bombSprite = new ImageProxy ( " Tnt . png " , s i z e , s i z e ) ;

64 w a l l S p r i te = new ImageProxy ( " Cobble . png " , s i z e , s i z e ) ;

65 ground Sprite = new ImageProxy ( " Grass . png " , s i z e , s i z e ) ;

66 powerup Sprite = new ImageProxy ( " Powerup . png " , s i z e , s i z e ) ;

67 t r a p S p r i te = new ImageProxy ( " Trap . png " , s i z e , s i z e ) ;

68 f i r e S p r i t e = new ImageProxy ( " Fi re . png " , s i z e , s i z e ) ;

69 p l a y e r S p r i te = new ImageProxy ( " Player . jpg " , 40 , 40 ) ;

70 }

71

72 p r i v a te boolean i n i t C o n n e c t i o n ( )

73 {

74 t h i s . network = new Network ( ) ;

75

76 i f ( ! t h i s . network . connect ( t h i s ) )

77 {

78 return f a l s e ;

79 }

80

81

82 }

return true ;

83

84 p r i v a te void runGame ( )

85 {

86 while (GLFW. glfwWindowShouldClose ( window ) != true )

87 {

88 readKeys ( ) ;

89 update Screen ( ) ;

90 waitFrame ( ) ;

91 }

92 }

93

94 p r i v a te boolean initWindow ( )

95 {

96 i f (GLFW. g l f w I n i t ( ) != true )

97 {

98 return f a l s e ;

99 }

100

101 window = GLFW. glfwCreateWindow (SCREEN\_LENGTH, SCREEN\_WIDTH, " Window " , 0 , 0 ) ;

102

103 GLFW. glfwShowWindow ( window ) ;

104 GLFW. glfw Make ContextCurrent ( window ) ;

105 GL. c r e a t e C a p a b i l i t i e s ( ) ;

106

107 // I n i t GL

108 // enable te x tu r e s s i n c e we ’ re going to use th e s e f o r our s p r i t e s

109 GL11 . gl Enable ( GL11 .GL\_TEXTURE\_2D) ;

110 GL11 . gl Enable ( GL11 .GL\_BLEND) ;

111 GL11 . glDepthFunc ( GL11 .GL\_ALWAYS) ;

112 GL11 . gl Blend Func ( GL11 .GL\_SRC\_ALPHA, GL11 .GL\_ONE\_MINUS\_SRC\_ALPHA) ;

113 // d i s a b l e the OpenGL depth t e s t s i n c e we ’ re r e n d e r i n g 2D g r a p h i c s

114 GL11 . gl Enable ( GL11 .GL\_DEPTH\_TEST) ;

115 GL11 . glMatrixMode ( GL11 .GL\_PROJECTION) ;

116 GL11 . g l L o a d Id e n t i ty ( ) ;

117 GL11 . gl Ortho ( 0 , SCREEN\_LENGTH, 0 , SCREEN\_WIDTH, - 1 , 1 ) ;

118 GL11 . glMatrixMode ( GL11 .GL\_MODELVIEW) ;

119 return true ;

120 }

121 p r i v a te void update Screen ( )

122 {

123 GLFW. g l fw Poll Events ( ) ;

124 GL11 . g l C l e a r ( GL11 .GL\_COLOR\_BUFFER\_BIT | GL11 .GL\_DEPTH\_BUFFER\_BIT) ;

125 GL11 . gl Enable ( GL11 .GL\_TEXTURE\_2D) ;

126 render Objects ( board ) ;

|  |  |  |
| --- | --- | --- |
| 127 |  | r e n d e r Pl a y e r s ( ) ; |
| 128 |  | GL11 . g l D i s a b l e ( GL11 .GL\_TEXTURE\_2D) ; |
| 129 |  | render Text ( ) ; |
| 130 |  | GLFW. glfw Swap Buffers ( window ) ; |
| 131 | } |  |
| 132 |  |  |
| 133 | p r i v a te void render Text ( ) | |
| 134 | { | |
| 135 | Text . draw String ( " S p e l l " + t h i s . main Player . skill Name + " Cooldown  " + ( i n t ) ( t h i s . main Player . skill Cooldown / 60 ) , 5 , 2 , 40 , 2 ) ; | |

136

Text . draw String ( " Player " + t h i s . main Player . id + " Health " + t h i s .

main Player . health , 45 , 2 , 40 , 2 ) ;

i n t s i z e = 4 0 ;

i f ( t h i s . main Player . health < 1 ) {

Text . draw String ( "GG" , 2 , 6 , 100 , 20 ) ;

}

f o r ( S i m p l i f i e d P l a y e r mpPlayer : p l a y e r s . v a l u e s ( ) )

{

i f ( t h i s . main Player . id != mpPlayer . id ) {

i f ( mpPlayer . c o o r d i n a te != n u l l )

{

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154 f l o a t x = mpPlayer . c o o r d i n a te . x\*4/ s i z e ;

155 f l o a t y = mpPlayer . c o o r d i n a te . y\*4/ s i z e ;

156

157 }

158

159 }

160

161 }

162

163 i f ( t h i s . d i s p l a y S t r i n g != n u l l )

164 {

165 Text . draw String ( t h i s . d i s p l a y S tr i n g , 10 , 40 , 55 , 4 ) ;

166 }

167

168

169 }

170

171 // bad design

172 p r i v a te void waitFrame ( )

173 {

174 try

175 {

176 // around 120 tiems a second

177 Thread . s l e e p ( 1L) ;

178 }

179 catch ( Interrupted Exception e )

180 {

181 // TODO Auto - generated catch block

182 e . print Stack Trace ( ) ;

183 }

184 }

185 p r i v a te void r e n d e r Pl a y e r s ( )

186 {

187 f o r ( C l i e n t Pl a y e r mpPlayer : p l a y e r s . v a l u e s ( ) ) {

188 mpPlayer . g e t S p r i te ( ) . drawImage ( ( i n t ) mpPlayer . c o o r d i n a te . x , ( i n t ) mpPlayer . c o o r d i n a te . y , 0 f , ( f l o a t ) ( 255 / 255 ) , ( f l o a t ) ( 127 / 255 ) ) ;

189 }

190 }

191

192

193 p r i v a te void render Objects ( Simplified Game Board board )

194 {

195 GL11 . gl Enable ( GL11 .GL\_BLEND) ;

196 GL11 . gl Blend Func ( GL11 .GL\_SRC\_ALPHA, GL11 .GL\_ONE\_MINUS\_SRC\_ALPHA) ;

197 i f ( board == n u l l )

198 {

199 return ;

200 }

201 i n t s i ze X = SCREEN\_LENGTH / board . g r i d S i z e ;

202 i n t s i ze Y = SCREEN\_WIDTH / board . g r i d S i z e ;

203

204

205 f o r ( i n t i = 0 ; i < board . g r i d S i z e ; i++) {

206 f o r ( i n t j = 0 ; j < board . g r i d S i z e ; j++) {

207 f l o a t red = ( f l o a t ) I n te g e r . value Of ( board . o b j e c t s [ i ] [ j ] . c o l o r . s u b s tr i n g ( 1 , 3 ) , 16 ) / 255 ;

208 f l o a t green = ( f l o a t ) I n te g e r . value Of ( board . o b j e c t s [ i ] [ j ] . c o l o r . s u b s tr i n g ( 3 , 5 ) , 16 ) / 255 ;

209 f l o a t blue = ( f l o a t ) I n te g e r . value Of ( board . o b j e c t s [ i ] [ j ] . c o l o r . s u b s tr i n g ( 5 , 7 ) , 16 ) / 255 ;

210 ImageProxy drawer ;

211 switch ( board . o b j e c t s [ i ] [ j ] . type )

212 {

213 d e f a u l t :

214 case GROUND:

215 drawer = new ImageProxy ( ground Sprite ) ;

216 break ;

217 case WALL:

218 drawer = new ImageProxy ( w a l l S p r i te ) ;

219 break ;

220 case TRAP:

221 drawer = new ImageProxy ( t r a p S p r i te ) ;

222 break ;

223 case POWERUP:

224 drawer = new ImageProxy ( powerup Sprite ) ;

225 break ;

226 case BOMB:

227 drawer = new ImageProxy ( bombSprite ) ;

228 break ;

229 }

230

231 drawer . drawImage ( i \* size X , j \* size Y , red , green , blue ) ;

232 GL11 . g l D i s a b l e ( GL11 .GL\_BLEND) ;

233 i f ( board . o b j e c t s [ i ] [ j ] . explode Animation == true )

234 {

235 GL11 . gl Blend Func ( GL11 .GL\_SRC\_ALPHA, GL11 . GL\_ONE\_MINUS\_SRC\_ALPHA) ;

236

drawer = new ImageProxy ( f i r e S p r i t e ) ;

drawer . drawImage ( i \* size X , j \* size Y , 1 , 1 , 1 ) ;

}

}

}

}

237

238

239

240

241

242

243

244

245

|  |  |  |
| --- | --- | --- |
| 246 | p r i v a te void readKeys ( ) |  |
| 247 | { |
| 248 |  |
| 249 | //TODO should read from events i n s te a d o f p o l l i n g |
| 250 | // https ://[www.](http://www/) glfw . org / docs / 3 . 3 / input\_guide . html |
| 251 |  |
| 252 |  |
| 253 |  |
| 254 |  |
| 255 | i f (GLFW. glfwGetKey ( t h i s . window , GLFW.GLFW\_KEY\_W) | == GLFW.GLFW\_TRUE) |
| 256 | { |  |
| 257 | i f ( ! t h i s . main Player . is Holding Up ) |  |
| 258 | { |  |
| 259 | t h i s . network . send PacketButton Press Up ( ) ; |  |
| 260 | } |  |
| 261 | } |  |
| 262 | e l s e |  |
| 263 | { |  |
| 264 | i f ( t h i s . main Player . is Holding Up ) |  |

|  |  |  |
| --- | --- | --- |
| 265 | { |  |
| 266 | t h i s . network . send PacketButton Release Up ( ) ; |  |
| 267 | } |  |
| 268 |  |  |
| 269 | } |  |
| 270 |  |  |
| 271 | i f (GLFW. glfwGetKey ( t h i s . window , GLFW.GLFW\_KEY\_S) | == GLFW.GLFW\_TRUE) |
| 272 | { |  |
| 273 | i f ( ! t h i s . main Player . isHoldingDown ) |  |
| 274 | { |  |
| 275 | t h i s . network . sendPacketButtonPressDown ( ) ; |  |
| 276 | } |  |
| 277 | } |  |
| 278 | e l s e |  |
| 279 | { |  |
| 280 | i f ( t h i s . main Player . isHoldingDown ) |  |
| 281 | { |  |
| 282 | t h i s . network . sendPacketButtonReleaseDown ( ) ; |  |
| 283 | } |  |
| 284 |  |  |
| 285 | } |  |
| 286 |  |  |
| 287 | i f (GLFW. glfwGetKey ( t h i s . window , GLFW.GLFW\_KEY\_A) | == GLFW.GLFW\_TRUE) |
| 288 | { |  |
| 289 | i f ( ! t h i s . main Player . i s H o l d i n g L e f t ) |  |
| 290 | { |  |
| 291 | t h i s . network . send Packet Button Press Left ( ) ; |  |
| 292 | } |  |
| 293 | } |  |

|  |  |  |
| --- | --- | --- |
| 294 | e l s e |  |
| 295 | { |  |
| 296 | i f ( t h i s . main Player . i s H o l d i n g L e f t ) |  |
| 297 | { |  |
| 298 | t h i s . network . send Packet Button Release Left ( ) ; |  |
| 299 | } |  |
| 300 |  |  |
| 301 | } |  |
| 302 |  |  |
| 303 | i f (GLFW. glfwGetKey ( t h i s . window , GLFW.GLFW\_KEY\_D) | == GLFW.GLFW\_TRUE) |
| 304 | { |  |
| 305 | i f ( ! t h i s . main Player . i s Holding Right ) |  |
| 306 | { |  |
| 307 | t h i s . network . send Packet Button Press Right ( ) ; |  |
| 308 | } |  |
| 309 | } |  |
| 310 | e l s e |  |
| 311 | { |  |
| 312 | i f ( t h i s . main Player . i s Holding Right ) |  |
| 313 | { |  |
| 314 | t h i s . network . send Packet Button Release Right ( ) ; |  |
| 315 | } |  |
| 316 |  |  |
| 317 | } |  |
| 318 |  |  |
| 319 |  |  |
| 320 | i f (GLFW. glfwGetKey ( t h i s . window , GLFW.GLFW\_KEY\_E) | == GLFW.GLFW\_TRUE) |
| 321 | { |  |
| 322 | i f ( ! t h i s . main Player . i s Holding Use ) |  |

323 {

324 t h i s . network . send PacketButton Press Use ( ) ;

325 }

326 }

327 e l s e

328 {

329 i f ( t h i s . main Player . i s Holding Use )

330 {

331 t h i s . network . send PacketButton Release Use ( ) ;

332 }

333 }

334

335 i f (GLFW. glfwGetKey ( t h i s . window , GLFW.GLFW\_KEY\_SPACE) == GLFW. GLFW\_TRUE)

336 {

337 i f ( ! t h i s . main Player . i s H o l d i n g S k i l l )

338 {

339 t h i s . network . send Packet Button Press Skill ( ) ;

340 }

341 }

342 e l s e

343 {

344 i f ( t h i s . main Player . i s H o l d i n g S k i l l )

345 {

346 t h i s . network . send Packet Button Release Skill ( ) ;

347 }

348 }

349

350 i f (GLFW. glfwGetKey ( window , GLFW.GLFW\_KEY\_ESCAPE) == GLFW.GLFW\_TRUE

)

351 {

352 GLFW. glfw SetWindow Should Close ( window , true ) ;

353 }

354 }

355

356 @Override

357 p u b l i c void update Player ( S i m p l i f i e d P l a y e r p l a y e r )

358 {

359 i f ( t h i s . main Player . id == p l a y e r . id )

360 {

361 C l i e n t Pl a y e r c l i e n t P l a y e r = new C l i e n t Pl a y e r ( player , p l a y e r S p r i te

, true ) ;

362 t h i s . main Player = p l a y e r ;

363 t h i s . p l a y e r s . put ( p l a y e r . id , c l i e n t P l a y e r ) ;

364 }

365 e l s e

366 {

367 C l i e n t Pl a y e r c l i e n t P l a y e r = new C l i e n t Pl a y e r ( player , p l a y e r S p r i te

, f a l s e ) ;

368 t h i s . p l a y e r s . put ( p l a y e r . id , c l i e n t P l a y e r ) ;

369 }

370

371 }

372 @Override

373 p u b l i c void add Player ( i n t id , S i m p l i f i e d P l a y e r p l a y e r )

374 {

375 i f ( p l a y e r . id == main Player . id )

376 {

377 C l i e n t Pl a y e r c l i e n t P l a y e r = new C l i e n t Pl a y e r ( player , p l a y e r S p r i te

, true ) ;

378 p l a y e r s . put ( id , c l i e n t P l a y e r ) ;

379 }

380 e l s e

381 {

382 C l i e n t Pl a y e r c l i e n t P l a y e r = new C l i e n t Pl a y e r ( player , p l a y e r S p r i te

, f a l s e ) ;

383 p l a y e r s . put ( id , c l i e n t P l a y e r ) ;

384 }

385 }

386

387 @Override

388 p u b l i c void remove Player ( i n t id )

389 {

390 p l a y e r s . remove ( id ) ;

391 }

392

393 @Override

394 p u b l i c void updateBoard ( Simplified Game Board gameboard )

395 {

396 i f ( t h i s . board == n u l l )

397 {

398 t h i s . i n i t P r o x i e s ( gameboard . s i z e / gameboard . g r i d S i z e ) ;

399 }

400 t h i s . board = gameboard ;

401

402 }

403

|  |  |  |
| --- | --- | --- |
| 404 |  |  |
| 405 | @Override |
| 406 | p u b l i c void update Display String ( S tr i n g | te x t ) |
| 407 | { |  |
| 408 | t h i s . d i s p l a y S t r i n g = te x t ; |  |
| 409 |  |  |
| 410 | } |  |
| 411 |  |  |
| 412 | } |  |
|  |  |  |
| 1 | package c l i e n t ; |  |
| 2 |  |  |
| 3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | import java . awt . image . Buffered Image ; import java . i o . F i l e ;  import java . i o . IOException ; import java . u t i l . HashMap ;  import javax . imageio . ImageIO ;  import org . l w j g l . glfw .GLFW; import org . l w j g l . opengl .GL; import org . l w j g l . opengl . GL11 ;  import shared . S i m p l i f i e d P l a y e r ;  p u b l i c c l a s s ProxyTest {  p u b l i c s t a t i c void main ( S tr i n g [ ] args ) { | |

20 i f (GLFW. g l f w I n i t ( ) != true )

21 {

22 return ;

23 }

24

25 long window ;

26 i n t SCREEN\_LENGTH = 1000 ;

27 i n t SCREEN\_WIDTH = 1000 ;

28 // I n i t Window

29 window = GLFW. glfwCreateWindow (SCREEN\_LENGTH, SCREEN\_WIDTH, " Window " , 0 , 0 ) ;

30

31 GLFW. glfwShowWindow ( window ) ;

32 GLFW. glfw Make ContextCurrent ( window ) ;

33 GL. c r e a t e C a p a b i l i t i e s ( ) ;

34

35 // I n i t GL

36 // enable te x tu r e s s i n c e we ’ re going to use th e s e f o r our s p r i t e s

37 GL11 . gl Enable ( GL11 .GL\_TEXTURE\_2D) ;

38 GL11 . gl Enable ( GL11 .GL\_BLEND) ;

39 GL11 . glDepthFunc ( GL11 .GL\_ALWAYS) ;

40 GL11 . gl Blend Func ( GL11 .GL\_SRC\_ALPHA, GL11 .GL\_ONE\_MINUS\_SRC\_ALPHA) ;

41 // d i s a b l e the OpenGL depth t e s t s i n c e we ’ re r e n d e r i n g 2D g r a p h i c s

42 GL11 . gl Enable ( GL11 .GL\_DEPTH\_TEST) ;

43 GL11 . glMatrixMode ( GL11 .GL\_PROJECTION) ;

44 GL11 . g l L o a d Id e n t i ty ( ) ;

45 GL11 . gl Ortho ( 0 , SCREEN\_LENGTH, 0 , SCREEN\_WIDTH, - 1 , 1 ) ;

46 GL11 . glMatrixMode ( GL11 .GL\_MODELVIEW) ;

47

48

49

50 S tr i n g path = " Test . jpg " ;

51

52 System . out . p r i n t l n ( "=== S i n g l e Image ===" ) ;

53 long s tart P = System . c u r r e n t Ti m e M i l l i s ( ) ;

54 Texture Loader l o a d e r = new Texture Loader ( ) ;

55 S p r i te newsprite = new S p r i te ( loader , path , 5 0 , 5 0 ) ;

56

57 long stopP = System . c u r r e n t Ti m e M i l l i s ( ) ;

58

59 System . out . p r i n t l n ( " One image loaded in : " + ( stopP - s tart P ) + " ms" ) ;

60

61 System . gc ( ) ; // Runtime . getRuntime ( ) . gc ( ) ;

62

63 System . out . p r i n t l n ( " \n=== Heavy Images == - show one image only

====" ) ;

64 long s ta r t R e a l = System . c u r r e n t Ti m e M i l l i s ( ) ;

65 Image File h1 = new Heavy Image File ( path , 5 0 , 5 0 ) ;

66 Image File h2 = new Heavy Image File ( path , 5 0 , 5 0 ) ;

67 Image File h3 = new Heavy Image File ( path , 5 0 , 5 0 ) ;

68 h2 . drawImage ( 0 , 0 , 1 , 1 , 1 ) ;

69 long stop Real = System . c u r r e n t Ti m e M i l l i s ( ) ;

70 System . out . p r i n t l n ( " Real loaded in " + ( stop Real - s ta r t R e a l ) + " ms" ) ;

71

72 System . gc ( ) ; // Runtime . getRuntime ( ) . gc ( ) ;

73

74 System . out . p r i n t l n ( " \n=== Proxy Images - show one image only ===" ) ;

75 long start Proxy = System . c u r r e n t Ti m e M i l l i s ( ) ;

76 Image File p1 = new ImageProxy ( path , 5 0 , 5 0 ) ;

77 Image File p2 = new ImageProxy ( path , 5 0 , 5 0 ) ;

78 Image File p3 = new ImageProxy ( path , 5 0 , 5 0 ) ;

79 p2 . drawImage ( 0 , 0 , 1 , 1 , 1 ) ;

80 long stop Proxy = System . c u r r e n t Ti m e M i l l i s ( ) ;

81 System . out . p r i n t l n ( " Proxy loaded in " + ( stop Proxy - start Proxy ) + " ms" ) ;

82

83 System . gc ( ) ; // Runtime . getRuntime ( ) . gc ( ) ;

84

85 System . out . p r i n t l n ( " \n=== Proxy Images - run a l l ===" ) ;

86 long startProxy A = System . c u r r e n t Ti m e M i l l i s ( ) ;

87 Image File a1 = new ImageProxy ( path , 5 0 , 5 0 ) ;

88 Image File a2 = new ImageProxy ( path , 5 0 , 5 0 ) ;

89 Image File a3 = new ImageProxy ( path , 5 0 , 5 0 ) ;

90 a1 . drawImage ( 0 , 0 , 1 , 1 , 1 ) ; ;

91 a2 . drawImage ( 0 , 0 , 1 , 1 , 1 ) ; ;

92 a3 . drawImage ( 0 , 0 , 1 , 1 , 1 ) ; ;

93 long stopProxyA = System . c u r r e n t Ti m e M i l l i s ( ) ;

94 System . out . p r i n t l n ( " Proxy loaded in " + ( stopProxyA - startProxy A )

+ " ms" ) ;

95 }

96

97 }

### Memento

### Kilo idėja į žaidimą įdėti įgūdį, kuris leidžia vartotojui “grįžti laiku” ir nekreipti dėmesio į klaidas, kurias padarė. Šiai idėjai tikslingai pravertė memento šablonas.

### UML diagrama:

### Text Description automatically generated with medium confidence

### Kodas:

2

3 import shared . Vector 2 f ;

4

5 p u b l i c c l a s s Memento {

6

7 i n t id ;

8 Vector 2 f c o o r d i n a te ;

9 boolean placedBomb ;

10 i n t health ;

11 i n t base Health ;

12 f l o a t speed ;

13 f l o a t base Speed ;

14 i n t s i z e ;

15 i n t death Counter ;

16

17

18

19 p u b l i c Memento( i n t id ,

20 Vector 2 f coordinate ,

21 boolean placedBomb , i n t health ,

22 i n t base Health , f l o a t speed , f l o a t baseSpeed ,

23 i n t s i z e , i n t bombCount , i n t bombTimer , i n t death Counter )

24 {

25 t h i s . id = id ;

26 t h i s . c o o r d i n a te = c o o r d i n a te ;

27 t h i s . placedBomb = placedBomb ;

28 t h i s . health = health ;

29 t h i s . base Health = base Health ;

30 t h i s . speed = speed ;

31

32

33

34 }

t h i s . base Speed = base Speed ; t h i s . s i z e = s i z e ;

t h i s . death Counter = death Counter ;

35

36 p u b l i c boolean g e t Sta te ( Pl a y e r In f o org ) {

37 i f ( id == org . get Id ( ) )

38 {

39 org . set Coordinate ( c o o r d i n a te ) ;

40 org . setPlacedBomb ( placedBomb ) ;

41 org . set Health ( health ) ;

42 org . set Base Health ( base Health ) ;

43 org . set Speed ( base Speed ) ;

44 org . set Base Speed ( base Speed ) ;

45 org . s e t S i z e ( s i z e ) ;

46

47

48 }

return true ;

49 return f a l s e ;

50

51 }

52

53 }

1

package s e r v e r ; import java . u t i l . \* ;

2

3

4

5

6 p u b l i c c l a s s Caretaker {

7

8 Array List<Memento> s t a t e s L i s t ;

9

10 p u b l i c Caretaker ( ) {

11 s t a t e s L i s t = new Array List<Memento>() ;

12 }

13

14 p u b l i c void add ( Memento s ta te ) {

15 s t a t e s L i s t . add ( s ta te ) ;

16 }

17

18 p u b l i c Memento get ( i n t index ) {

19 Memento r e s t o r e S t a t e = s t a t e s L i s t . get ( index ) ;

20 s t a t e s L i s t . remove ( index ) ;

21 return r e s t o r e S t a t e ;

22 }

23

24 p u b l i c Memento undo ( )

25 {

26 // popping l a s t s ta te

27 i n t index = s t a t e s L i s t . s i z e ( ) - 1 ;

28 Memento r e s t o r e S t a t e = s t a t e s L i s t . get ( index ) ;

29 s t a t e s L i s t . remove ( index ) ;

30 return r e s t o r e S t a t e ;

31 }

32

33

34 p u b l i c i n t s i z e ( ) {

35

return s t a t e s L i s t . s i z e ( ) ;

}

}

36

37

38

1 package s e r v e r ;

2

3 import shared . PacketUpdate Player Pos ;

4 import shared . Vector 2 f ;

5

6 import java . u t i l . Random ;

7

8 p u b l i c c l a s s Pl a y e r In f o

9 {

10 i n t id ;

11 Vector 2 f c o o r d i n a te ;

12 boolean i s H o l d i n g L e f t ;

13 boolean i s Holding Right ;

14 boolean is Holding Up ;

15 boolean isHoldingDown ;

16 boolean i s Holding Use ;

17 boolean i s H o l d i n g S k i l l ;

18 boolean i s Holding Pause ;

19 boolean is Holding Un Pause ;

20 boolean placedBomb = true ;

21 i n t health ;

22 i n t base Health ;

23 f l o a t speed ;

24 f l o a t base Speed ;

25 i n t s i z e ;

26 i n t bombCount ;

27 i n t bombTimer ;

28

29 i n t death Counter ;

30

31 S k i l l A l g o r i th m s k i l l A l g o r i th m ;

32

33 boolean canDie = true ;

34

35 p u b l i c Pl a y e r In f o ( )

36 {

37

38 death Counter = 0 ;

39

40 Random r = new Random ( ) ;

41 i n t low = 100 ;

42 i n t high = 800 ;

43

44 setBombTimer ( 1 ) ;

45

46

47

48 i n t x = r . next Int ( high - low ) + low ;

49 i n t y = r . next Int ( high - low ) + low ;

50

51 t h i s . base Health = 2 ;

52 t h i s . c o o r d i n a te = new Vector 2 f ( ) ;

53 t h i s . c o o r d i n a te . x = x ;

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68 }

t h i s . c o o r d i n a te . y = y ;

t h i s . i s H o l d i n g L e f t = f a l s e ; t h i s . i s Holding Right = f a l s e ; t h i s . is Holding Up = f a l s e ; t h i s . isHoldingDown = f a l s e ; t h i s . i s Holding Pause = f a l s e ;

t h i s . is Holding Un Pause = f a l s e ;

t h i s . s k i l l A l g o r i th m = new Going Back In Time Skill ( ) ; t h i s . s i z e = 4 0 ;

t h i s . speed = 5 f ;

t h i s . base Speed = 5 f ; t h i s . health = 2 ;

t h i s . bombCount = 2 ;

// t h i s . p l a y e r S ta ts = new Concrete Player ( ) ;

69

70 p u b l i c void setBombTimer ( i n t time ) {

71 bombTimer = 30\* time ;

72 }

73

74

75 p u b l i c void reduce Timer ( ) {

76 bombTimer - - ;

77 }

78

79

80 p u b l i c void Change Player Location ( i n t x , i n t y ) {

81

82 t h i s . c o o r d i n a te . x = x ;

83 t h i s . c o o r d i n a te . y = y ;

84

85 }

86

87 p u b l i c Pl a y e r In f o ( i n t id , Vector 2 f c o o r d i n a te )

88 {

89 t h i s . id = id ;

90 t h i s . c o o r d i n a te = c o o r d i n a te ;

91 t h i s . i s H o l d i n g L e f t = f a l s e ;

92 t h i s . i s Holding Right = f a l s e ;

93 t h i s . is Holding Up = f a l s e ;

94 t h i s . isHoldingDown = f a l s e ;

95 t h i s . i s Holding Use = f a l s e ;

96 t h i s . i s Holding Pause = f a l s e ;

97 t h i s . is Holding Un Pause = f a l s e ;

98 t h i s . s i z e = 4 0 ;

99 t h i s . speed = 2 . 5 f ;

100 t h i s . base Speed = 2 . 5 f ;

101 t h i s . health = 3 ;

102 t h i s . bombCount = 2 ;

103 // Test

104 }

105

106 p u b l i c S k i l l A l g o r i th m g e t S k i l l A l g o r i th m ( )

107 {

108 return s k i l l A l g o r i th m ;

109 }

110

111 p u b l i c void s e t S k i l l A l g o r i th m ( S k i l l A l g o r i th m s k i l l A l g o r i th m )

112 {

113 t h i s . s k i l l A l g o r i th m = s k i l l A l g o r i th m ;

114 }

115

116

117 p u b l i c void onTick ( )

118 {

119 t h i s . s k i l l A l g o r i th m . onTick ( t h i s ) ;

120 }

121

122 p u b l i c void tr y U s i n g S p e l l ( )

123 {

124 t h i s . s k i l l A l g o r i th m . u s e S k i l l ( t h i s ) ;

125 }

126

127 p u b l i c i n t getCooldown ( )

128 {

129 return t h i s . s k i l l A l g o r i th m . getCooldown ( ) ;

130 }

131

132 p u b l i c S tr i n g getName ( )

133 {

134 return t h i s . s k i l l A l g o r i th m . getName ( ) ;

135 }

136

137 p u b l i c PacketUpdate Player Pos getPacketUpdate Player Pos ( i n t id )

138 {

139 PacketUpdate Player Pos newPack = new PacketUpdate Player Pos ( ) ;

140 newPack . id = id ;

|  |  |  |
| --- | --- | --- |
| 141 | newPack . c o o r d i n a te = t h i s . c o o r d i n a te ; |  |
| 142 | newPack . i s H o l d i n g L e f t = t h i s . i s H o l d i n g L e f t ; |
| 143 | newPack . i s Holding Right = t h i s . i s Holding Right ; |
| 144 | newPack . is Holding Up = t h i s . is Holding Up ; |
| 145 | newPack . isHoldingDown = t h i s . isHoldingDown ; |
| 146 | newPack . i s Holding Use = t h i s . i s Holding Use ; |
| 147 | newPack . s i z e = t h i s . s i z e ; |
| 148 | newPack . i s H o l d i n g S k i l l = t h i s . i s H o l d i n g S k i l l ; |
| 149 | newPack . i s Holding Pause = t h i s . i s Holding Pause ; |
| 150 | newPack . is Holding Un Pause = t h i s . is Holding Un Pause ; |
| 151 | newPack . skill Name = t h i s . getName ( ) ; |
| 152 | newPack . skill Cooldown = t h i s . getCooldown ( ) ; |
| 153 | newPack . health = t h i s . health ; |
| 154 | return newPack ; |
| 155 | } |
| 156 |  |
| 157 | p u b l i c void r e s t o r e S t a t e ( Memento memento ) { |
| 158 |  |
| 159 | i f ( memento . g e t Sta te ( t h i s ) ) { |
| 160 | System . out . p r i n t l n ( " S u c c e s s f u l l y r e s to r e d s ta te " ) ; |
| 161 | } e l s e { |
| 162 | System . out . p r i n t l n ( " Unable to r e s t o r e f o r Caretaker | " + t h i s . id ) ; |
| 163 | } |  |
| 164  165  166  167  168 | }  p u b l i c Memento save State ( ) {  return new Memento( id , coordinate , placedBomb , health , base Health , speed , baseSpeed , s i z e , bombCount , bombTimer , death Counter ) ; | |

169 }

170

171 p u b l i c i n t get Id ( ) {

172 return id ;

173 }

174

175 p u b l i c void s e t I d ( i n t id ) {

176 t h i s . id = id ;

177 }

178

179 p u b l i c Vector 2 f get Coordinate ( ) {

180 return c o o r d i n a te ;

181 }

182

183 p u b l i c void set Coordinate ( Vector 2 f c o o r d i n a te ) {

184 t h i s . c o o r d i n a te = new Vector 2 f ( c o o r d i n a te . x , c o o r d i n a te . y ) ;

185 }

186

187 p u b l i c boolean isPlacedBomb ( ) {

188 return placedBomb ;

189 }

190

191 p u b l i c void setPlacedBomb ( boolean placedBomb ) {

192 t h i s . placedBomb = placedBomb ;

193 }

194

195 p u b l i c i n t get Health ( ) {

196 return health ;

197 }

198

199 p u b l i c void set Health ( i n t health ) {

200 t h i s . health = health ;

201 }

202

203 p u b l i c i n t get Base Health ( ) {

204 return base Health ;

205 }

206

207 p u b l i c void set Base Health ( i n t base Health ) {

208 t h i s . base Health = base Health ;

209 }

210

211 p u b l i c f l o a t getSpeed ( ) {

212 return speed ;

213 }

214

215 p u b l i c void set Speed ( f l o a t speed ) {

216 t h i s . speed = speed ;

217 }

218

219 p u b l i c f l o a t getBase Speed ( ) {

220 return base Speed ;

221 }

222

223 p u b l i c void set Base Speed ( f l o a t base Speed ) {

224 t h i s . base Speed = base Speed ;

225 }

226

227 p u b l i c i n t g e t S i z e ( ) {

228 return s i z e ;

229 }

230

231 p u b l i c void s e t S i z e ( i n t s i z e ) {

232 t h i s . s i z e = s i z e ;

233 }

234

235 p u b l i c i n t getBombCount ( ) {

236 return bombCount ;

237 }

238

239 p u b l i c void setBombCount ( i n t bombCount) {

240 t h i s . bombCount = bombCount ;

241 }

242

243 p u b l i c i n t getDeath Counter ( ) {

244 return death Counter ;

245 }

246

247 p u b l i c void setDeath Counter ( i n t death Counter ) {

248 t h i s . death Counter = death Counter ;

249 }

250

251 p u b l i c i n t getBombTimer ( ) {

252 return bombTimer ;

253 }

254

255 }

### Interpreter

### Timeline Description automatically generated

Šis šablonas buno naudingas pagyvinti žaidimą naudojant console komandas

Interpreter

1 package s e r v e r ;

2

3 import java . u t i l . Random ;

4

5 p u b l i c c l a s s Place Des Wall extends Expression {

6 @Override

7 p u b l i c GameBoard randomly ( i n t quantity , GameBoard board ) {

8

9 Random r = new Random ( ) ;

10 i n t low = 1 ;

11 i n t high = 1 9 ;

12

13

14 f o r ( i n t i = 0 ; i<quantity ; i++){

15 i n t x = r . next Int ( high - low ) + low ;

16 i n t y = r . next Int ( high - low ) + low ;

17

18 board . addObject ( x , y , " des Wall " ) ;

19

20 }

21

22

23

24 }

return board ;

25

26 @Override

27 p u b l i c GameBoard l i n e a r ( i n t quantity , GameBoard board ) {

28

29

30

31 f o r ( i n t i = 0 ; i<quantity ; i++){

32

33 i n t x = 9 ;

34 i n t y = 9 + i ;

35

36

37

38

39 board . addObject ( x , y , " des Wall " ) ;

40

41 }

42

43

44

45 }

return board ;

46

47 @Override

48 p u b l i c GameBoard z i g z a g ( i n t quantity , GameBoard board ) {

49

50

51 i n t k i e k i s = quantity ;

52

53

54

55

56

57 f o r ( i n t i = 1 ; i <20; i++){

58

i n t x = i ; i n t y = i ;

i f ( k i e k i s != 0 ) { k i e k i s - - ;

} e l s e

{

board . addObject ( x , y , " des Wall " ) ;

}

}

return board ;

}

}

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

1

package s e r v e r ;

import java . u t i l . Random ;

p u b l i c c l a s s Place Wall extends Expression {

2

3

4

5

6

7

8 @Override

9 p u b l i c GameBoard randomly ( i n t quantity , GameBoard board ) {

10

11 Random r = new Random ( ) ;

12 i n t low = 1 ;

13 i n t high = 1 9 ;

14

15

16 f o r ( i n t i = 0 ; i<quantity ; i++){

17 i n t x = r . next Int ( high - low ) + low ;

18 i n t y = r . next Int ( high - low ) + low ;

19

20

21 board . addObject ( x , y , " wall " ) ;

22

23

24 }

25

26

27

28 }

return board ;

29

30 @Override

31 p u b l i c GameBoard l i n e a r ( i n t quantity , GameBoard board ) {

32

33

34

35

36

37 f o r ( i n t i = 0 ; i<quantity ; i++){

38

39 i n t x = 9 + i ;

40 i n t y = 9 ;

41

42

43

44

45 board . addObject ( x , y , " wall " ) ;

46

47 }

48

49

50

51 }

return board ;

52

53 @Override

54 p u b l i c GameBoard z i g z a g ( i n t quantity , GameBoard board ) {

55

56

57 f o r ( i n t i = 0 ; i<quantity ; i++){

58

59 i n t x = 8 + i ;

60 i n t y = 8 + i ;

61

62

63

64 board . addObject ( x , y , " wall " ) ;

65

66

}

return board ;

}

}

67

68

69

70

71

72

1 package s e r v e r ;

2

3 import java . u t i l . Random ;

4

5 p u b l i c c l a s s Place Trap extends Expression {

6 @Override

7 p u b l i c GameBoard randomly ( i n t quantity , GameBoard board ) {

8

9 Random r = new Random ( ) ;

10 i n t low = 1 ;

11 i n t high = 1 9 ;

12

13

14

15 f o r ( i n t i = 0 ; i<quantity ; i++){

16 i n t x = r . next Int ( high - low ) + low ;

17 i n t y = r . next Int ( high - low ) + low ;

18

19 board . addObject ( x , y , " trap " ) ;

20

21 }

22

23

24

25 }

return board ;

26

27 @Override

28 p u b l i c GameBoard l i n e a r ( i n t quantity , GameBoard board ) {

29

30

31 f o r ( i n t i = 0 ; i<quantity ; i++){

32

33 i n t x = 4 + i \* 2 ;

34 i n t y = 9 ;

35

36

37

38

39 board . addObject ( x , y , " trap " ) ;

40

41 }

42

43

44

45 }

return board ;

46

47 @Override

48 p u b l i c GameBoard z i g z a g ( i n t quantity , GameBoard board ) {

49

50 i n t r e v e r s e = - 1 ;

51

f o r ( i n t i = 0 ; i<quantity ; i++){

i n t x = 4 + i ;

i n t y = 9 + ( i \* r e v e r s e ) ;

r e v e r s e \*= -1;

board . addObject ( x , y , " trap " ) ;

}

return board ;

}

}

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

1

package s e r v e r ;

p u b l i c a b s tr a c t c l a s s Expression {

p u b l i c a b s tr a c t GameBoard randomly ( i n t quantity , GameBoard board ) ; p u b l i c a b s tr a c t GameBoard l i n e a r ( i n t quantity , GameBoard board ) ; p u b l i c a b s tr a c t GameBoard z i g z a g ( i n t quantity , GameBoard board ) ;

2

3

4

5

6

7

8

9

10

11 }

1 package s e r v e r ;

2

3 import java . u t i l . Locale ;

4

5 p u b l i c c l a s s Conversion Context {

6

7 p r i v a te S tr i n g given Sentence = " " ;

8

9 p r i v a te S tr i n g r e s u l t = " " ;

10

11 p r i v a te S tr i n g what = " " ;

12 p r i v a te S tr i n g where = " " ;

13

14 p r i v a te i n t quantity = 0 ;

15

16 S tr i n g [ ] p a r ts ;

17

18 p u b l i c Conversion Context ( S tr i n g sentence ) {

19

20 given Sentence = sentence . toLowerCase ( ) ;

21

22 p a r ts = get Sentence ( ) . s p l i t ( " " ) ;

23

24 what = p a r ts [ 2 ] ;

25

26 quantity = I n te g e r . p a r s e I n t ( p a r ts [ 1 ] ) ;

27

28 where = p a r ts [ 3 ] ;

29

30 r e s u l t = given Sentence + " " ;

31

32 }

33

34 p u b l i c S tr i n g get Sentence ( ) {

35 return given Sentence ;

36 }

37

38 p u b l i c S tr i n g getWhat ( ) {

39 return what ;

40 }

41

42 p u b l i c S tr i n g getWhere ( ) {

43 return where ;

44 }

45

46 p u b l i c S tr i n g get Re sul t ( ) {

47 return r e s u l t ;

48 }

49

50 p u b l i c i n t get Quantity ( ) {

51 return quantity ;

52 }

53

54

55

56 }

### Chain of responsibility

Diagram

Description automatically generated

Šį šabloną pasirinkau, kad pagyvinti interpreter darbą

Chain of responsibility

1 package s e r v e r ;

2

3 p u b l i c c l a s s Task {

4 p r i v a te i n t quantity ;

5 p r i v a te GameBoard board ;

6 p r i v a te S tr i n g what ;

7 p r i v a te S tr i n g where ;

8

9 p u b l i c Task ( i n t quantity , S tr i n g what , S tr i n g where , GameBoard newboard ) {

10 t h i s . quantity = quantity ;

11 t h i s . what = what ;

12 t h i s . where = where ;

13 board = newboard ;

14 }

15

16

17

18 p u b l i c i n t get Quantity ( ) {

19 return quantity ;

20 }

21

22 p u b l i c S tr i n g getWhat ( ) {

23 return what ;

24 }

25

26 p u b l i c S tr i n g getWhere ( ) {

27 return where ;

|  |  |  |
| --- | --- | --- |
| 28  29  30 | }  p u b l i c GameBoard getBoard ( ) { | |
| 31 | return board ; | |
| 32 | } | |
| 33 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c i n t e r f a c e Chain { |  |
| 4 |  |  |
| 5 | p u b l i c void setNextChain ( Chain nextChain ) ; |  |
| 6 | p u b l i c GameBoard r e s u l t ( Task te x t ) ; |  |
| 7 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c c l a s s Generate Walls implements Chain { |  |
| 4 |  |  |
| 5 |  |  |
| 6 | p r i v a te Chain nextIn Chain ; |  |
| 7 |  |  |
| 8 | @Override |  |
| 9 | p u b l i c void setNextChain ( Chain nextChain ) | { |
| 10 | nextIn Chain = nextChain ; |  |
| 11 |  |  |
| 12 | } |  |
| 13 |  |  |
| 14 | @Override |  |
| 15 | p u b l i c GameBoard r e s u l t ( Task te x t ) { |  |

16

17 GameBoard r e s u l t = te x t . getBoard ( ) ;

18

19 i f ( te x t . getWhat ( ) . e q u a l s ( " wall " ) ) {

20

21 Place Wall newWalls = new Place Wall ( ) ;

22

23 System . out . p r i n t l n ( " Turiu s i e n a s s u k u r t i " ) ;

24

25 switch ( te x t . getWhere ( ) ) {

26 case " randomly " : r e s u l t = newWalls . randomly ( te x t . get Quantity ( ) , te x t . getBoard ( ) ) ;

27 break ;

28

29 case " z i g z a g " : r e s u l t = newWalls . z i g z a g ( te x t . get Quantity ( ) , te x t . getBoard ( ) ) ;

30 break ;

31

32 case " l i n e a r " : r e s u l t = newWalls . l i n e a r ( te x t . get Quantity ( ) , te x t . getBoard ( ) ) ;

33 break ;

34

35 }

36

37 return r e s u l t ;

38

39

40

41 } e l s e {

|  |  |  |
| --- | --- | --- |
| 42 | nextIn Chain . r e s u l t ( te x t ) ; |  |
| 43 | } |
| 44 |  |
| 45 |  |
| 46 |  |
| 47 | return r e s u l t ; |
| 48 |  |
| 49 | } |
| 50 | } |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c c l a s s GenerateTrap implements Chain { |  |
| 4 | p r i v a te Chain nextIn Chain ; |  |
| 5 |  |  |
| 6 | @Override |  |
| 7 | p u b l i c void setNextChain ( Chain nextChain ) | { |
| 8 | nextIn Chain = nextChain ; |  |
| 9 |  |  |
| 10 | } |  |
| 11 |  |  |
| 12 | @Override |  |
| 13 | p u b l i c GameBoard r e s u l t ( Task te x t ) { |  |
| 14 |  |  |
| 15 | GameBoard r e s u l t = te x t . getBoard ( ) ; |  |
| 16  17  18  19 | i f ( te x t . getWhat ( ) . e q u a l s ( " trap " ) ) {  Place Trap newWalls = new Place Trap ( ) ; | |

20

21 switch ( te x t . getWhere ( ) ) {

22 case " randomly " : r e s u l t = newWalls . randomly ( te x t . get Quantity ( ) , te x t . getBoard ( ) ) ;

23 break ;

24

25 case " z i g z a g " : r e s u l t = newWalls . z i g z a g ( te x t . get Quantity ( ) , te x t . getBoard ( ) ) ;

26 break ;

27

28 case " l i n e a r " : r e s u l t = newWalls . l i n e a r ( te x t . get Quantity ( ) , te x t . getBoard ( ) ) ;

29 break ;

30

31 }

32

33 return r e s u l t ;

34

35

36

37 } e l s e {

38 nextIn Chain . r e s u l t ( te x t ) ;

39 }

40

41

42

43 return r e s u l t ;

44

45 }

|  |  |  |
| --- | --- | --- |
| 46 |  |  |
| 47 | } |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c c l a s s GenerateDesWall implements Chain | { |
| 4 | p r i v a te Chain nextIn Chain ; |  |
| 5 |  |  |
| 6 | @Override |  |
| 7 | p u b l i c void setNextChain ( Chain nextChain ) | { |
| 8 | nextIn Chain = nextChain ; |  |
| 9 |  |  |
| 10 | } |  |
| 11 |  |  |
| 12 | @Override |  |
| 13 | p u b l i c GameBoard r e s u l t ( Task te x t ) { |  |
| 14 |  |  |
| 15 | GameBoard r e s u l t = te x t . getBoard ( ) ; |  |
| 16  17  18  19  20  21  22  23  24  25 | i f ( te x t . getWhat ( ) . e q u a l s ( " d e s w a l l " ) ) {  Place Des Wall newWalls = new Place Des Wall ( ) ; switch ( te x t . getWhere ( ) ) {  case " randomly " : r e s u l t = newWalls . randomly ( te x t . get Quantity ( ) , te x t . getBoard ( ) ) ;  break ;  case " z i g z a g " : r e s u l t = newWalls . z i g z a g ( te x t . | |

get Quantity ( ) , te x t . getBoard ( ) ) ;

26 break ;

27

28 case " l i n e a r " : r e s u l t = newWalls . l i n e a r ( te x t . get Quantity ( ) , te x t . getBoard ( ) ) ;

29 break ;

30

31 }

32

33 return r e s u l t ;

34

35

36

37 }

38

39

40

41 return r e s u l t ;

42

43 }

44 }

### Composite

### Diagram Description automatically generated with medium confidence

### 

Šį algoritmą pasirinkau, nes reikėjo būdo surinkti informaciją apie žaidimą ir ją pavaizduoti.

Composition

package s e r v e r ;

p u b l i c c l a s s Item extends ItemComponent{ S tr i n g itemName ;

S tr i n g itemDes ;

1

2

3

4

5

6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | i n t count ; | |  | | | | |
| 8 |  | |
| 9 | p u b l i c Item ( S tr i n g newName , S tr i n g | | newDesc , | | i n t | count ) { | |
| 10 | itemName = newName ; | |  | |  |  | |
| 11 | itemDes = newDesc ; | |  | |  |  | |
| 12 | t h i s . count = count ; | |  | |  |  | |
| 13 | } | |  | |  |  | |
| 14 |  | |  | |  |  | |
| 15 |  | |  | |  |  | |
| 16 | p u b l i c S tr i n g getItemName ( ) { | |  | |  |  | |
| 17 | return itemName ; | |  | |  |  | |
| 18 | } | |  | |  |  | |
| 19 |  | |  | |  |  | |
| 20 | p u b l i c S tr i n g getItem Des ( ) { | |  | |  |  | |
| 21 | return itemDes ; | |  | |  |  | |
| 22 | } | |  | |  |  | |
| 23 |  | |  | |  |  | |
| 24 | p u b l i c i n t getCount ( ) { | |  | |  |  | |
| 25 | return count ; | |  | |  |  | |
| 26 | } | |  | |  |  | |
| 27 |  | |  | |  |  | |
| 28  29 | p u b l i c void increase Count ( ) {  count++; | | | | | | |
| 30 | } |  | |  |  | |  |
| 31 |  |  | |  |  | |  |
| 32 | p u | b l i c void dis play I tem Inf ormation ( ) { | |  |  | |  |
| 33 |  | System . out . p r i n t l n ( itemName + " Desc : | | " | + itemDes + | | " Count : " |
|  | + count ) ; | | | | | | |
| 34 | } | | | | | | |

35

}

36

37

38

39

1 package s e r v e r ;

2

3 p u b l i c a b s tr a c t c l a s s ItemComponent {

4

5 p u b l i c void add ( ItemComponent newItemComponent ) {

6 throw new Unsupported Operation Exception ( ) ;

7 }

8

9 p u b l i c void remove ( ItemComponent newItemComponent ) {

10 throw new Unsupported Operation Exception ( ) ;

11 }

12

13 p u b l i c ItemComponent getComponent ( i n t index ) {

14 throw new Unsupported Operation Exception ( ) ;

15 }

16

17 p u b l i c S tr i n g getItemName ( ) {

18 throw new Unsupported Operation Exception ( ) ;

19 }

20

21 p u b l i c i n t getItemCount ( ) {

22 throw new Unsupported Operation Exception ( ) ;

23 }

24

p u b l i c void dis play I tem Inf ormation ( ) {

throw new Unsupported Operation Exception ( ) ;

}

}

25

26

27

28

29

30

31

1

package s e r v e r ;

import java . u t i l . Array List ; import java . u t i l . I t e r a t o r ;

p u b l i c c l a s s ItemGroup extends ItemComponent{ Array List itemComponents = new Array List ( ) ;

S tr i n g groupName ; S tr i n g groupDesc ;

p u b l i c ItemGroup ( S tr i n g newName , S tr i n g newDesc ) { groupName = newName ;

groupDesc = newDesc ;

}

p u b l i c S tr i n g getGroupName ( ) {

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22 }

return groupName ;

23

24 p u b l i c S tr i n g getGroupDesc ( ) {

25 return groupDesc ;

26 }

27

28 p u b l i c void add ( ItemComponent newItemComponent ) {

29 itemComponents . add ( newItemComponent ) ;

30 }

31

32 p u b l i c void remove ( ItemComponent newItemComponent ) {

33 itemComponents . remove ( newItemComponent ) ;

34 }

35

36 p u b l i c ItemComponent getComponent ( i n t index ) {

37

38

39 }

return ( ItemComponent ) itemComponents . get ( index ) ;

40

41

42 p u b l i c void dis play I tem Inf ormation ( ) {

43 System . out . p r i n t l n ( getGroupName ( ) + " \n " + getGroupDesc ( ) + " \ n " ) ;

44

45

46 I t e r a t o r i te m I te r a to r = itemComponents . i t e r a t o r ( ) ;

47

48 while ( i te m I te r a to r . hasNext ( ) ) {

|  |  |  |
| --- | --- | --- |
| 49  50  51  52  53  54  55  56 | ItemComponent i tem Info = ( ItemComponent ) i te m I te r a to r . next  ( ) ;  i tem Info . di splay I tem Inform ation ( ) ;  }  System . out . p r i n t l n ( ) ; System . out . p r i n t l n ( ) ;  } | |
| 57 |  |  |
| 58 |  |  |
| 59 |  |  |
| 60 |  |  |
| 61 | } |  |
|  |  |  |
| 1 | package s e r v e r ; |  |
| 2 |  |  |
| 3 | p u b l i c c l a s s Item List { |  |
| 4 |  |  |
| 5 | ItemComponent i te m L i s t ; |  |
| 6 |  |  |
| 7 | p u b l i c Item List ( ItemComponent | l i s t ) { |
| 8 | i te m L i s t = l i s t ; |  |
| 9 | } |  |
| 10 |  |  |
| 11 | p u b l i c void g e t I te m L i s t ( ) { |  |
| 12 | i te m L i s t . displ ay I tem Information ( ) ; | |
| 13 | } | |
| 14 |  | |

15 }