Lan Praktikoa : TETЯIS JOKOA

Juanan Pereira

Azken eguneraketa: 2015/11/01

Aurkibidea

OHARRA: 1-7 ariketa multzoko erantzunak foroan argitaratuko dira, beraz, horiek ez dituzu zertan egin behar ez baduzu nahi edo denborarik ez baduzu. Entregatu eta beraz ebaluatu egingo direnak 8., 9. eta 10. ariketak dira.

Sarrera

Tetris mundu mailan joku ospetsuenetariko bat da. Puzle moduko joko bat, Alexey Pajitnov programatzaileak 1984an garatu zuen, Rusian. Ordutik, ehunaka aldaera izan ditu. Praktika honetan gure Tetris jokoaren aldaera propioa garatuko dugu.

Jokoa 10x20 gelaxkaz osatutako taula huts batekin hasten da. Goi-ezkerreko gelaxkaren koordenatuak (0,0) eta behe-eskubikoak (9,19) izanik. Ikusten duzun moduan, y ardatza beheraka doa. Tetrisan, 7 pieza posible daude. Jokoa hasteko, bat aukeratzen da, ausaz, eta taularen goiko aldean margotzen da. Pieza hori automatikoki jaitsiko da, pausuz-pausu, beheko alderuntz.

Arau nagusiak

1. Pieza batek ezin du hartu beste pieza baten posizioa (piezak ezin dira teilakatu) eta ezin da taulatik kanpo mugitu.
2. Jaisten ari den pieza batek beste pieza batekin talka egiten duenean - edo taularen beheko aldea ikutzen duenean -, bere mugimendua eten egiten da eta taularen goiko aldetik beste pieza berri bat agertuko da.
3. Taularen beheko aldean geratu diren piezek lerroak sortzen dituzte. Lerro bat osoa edo partziala izan daiteke. Lerro oso bat sortzen bada (lerroko gelaxka guztiak piezez osatuta dago), taulatik desagertuko da eta bere gainean dauden bloke guztiak jaitsi egingo dira, beste bloke batzuk edo taularen beheko ertza ikutu arte.
4. Jokoan ezin bada pieza berri bat sartu (beheko lerro partzialak goraino igo direlako) jokoa amaitu egingo da (Game Over).

Erabiltzailearen mugimenduak

Erabiltzaileak teklatuko geziak erabili ditzake piezak mugitu eta biratzeko. Ezker, Eskubi, Behera geziak pieza norabide horretan mugitzen dute (gelaxka bat pultsazio bakoitzeko). Erabiltzaileak zuriune-barra ere erabili dezake. Horrek pieza berehala mugituko du taularen beheko alderaino edo beste pieza batekin talka egin arte. Pieza bat mugitu edo biratzerakoan, ezin da taularen mugak gainditu edo beste pieza batekin teilakatu.

Praktikaren diseinua

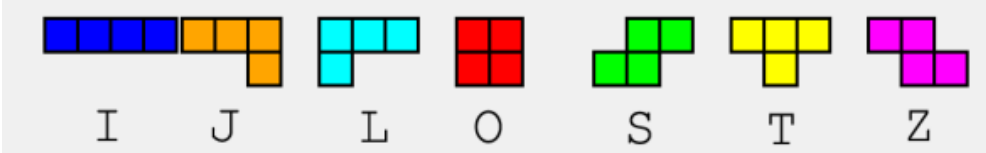
Praktikaren diseinuarekin laguntzeko, JavaScripten egindako txantiloi bat izango duzu. Bertan, programatu beharko dituzun klaseak agertzen dira, bakoitzaren metodoen zehaztapenekin (inplementatu beharko dituzunak). Pausuz-pausu joango gara, eta modu inkrementalan garatuko ditugu behar diren metodo nagusiak. Ikastaroak esleituta duen denbora mugak direla eta, metodo batzuk programatuta jasoko dituzu ikastaroan zehar.

Programatu behar dituzun metodoetan "ZURE KODEA HEMEN" katea aurkituko duzu. Komentario hori ezabatu eta horren ordez, dagokion kodea sartu behar duzu. Tetris-aren bertsio sinplifikatu batekin jolastu ahal izateko, metodo horiek guztiak programatu beharko dituzu.

Tetris klaseen ikuspegi orokorra

Ikastaroko webgunetik tetris.js fitxategia jaitsi eta aztertu ezazu. Oinarrizko klase batzuen goiburua jada bertan programatu ditugu:

- Point: oinarrizko klasea. Pantailan puntu baten (x,y) posizioa irudikatzeko. Programatuta dago jada.
- Rectangle: programatu beharko duzun lehenengo klasea. Laguntza gisa beharko dituzun metodo batzuk jada garatuta daude: metodo eraikitzailea, hasieratze metodoa, lerro baten zabalera esleitzeko metodoa eta laukizuzen baten atzealdeko kolorea esleitzeko metodoa. Gainontzeko metodoak zuk garatu beharko dituzu, laguntza gisa utzi diren iruzkin edo komentarioak esaten duten jarraituz.
- Shape: piezak margotzeko oinarrizko klasea. Tetrisean 7 pieza edo Shape ezberdin daude. Pieza bakoitza 4 koloredun-blokez osatuta dago. Pieza hauek Tetromino izenekin esagutzen dira baita ere eta horrela irudikatzen dira:



Shape bat eraiki ahal izateko, Point, Rectangle eta Block klaseak programatu beharko ditugu lehenengo eta behin.

- Block: esan bezala, gure Tetris jokoaren taulan 10x20 gelaxka daude non gelaxka bakoitza 30 pixel zabalerako lauki bat den. Bloke batek gelaxka bakar bat betetzen du. Ariketa honetan taula baten posizioa adierazteko, gelaxka baten koordenatuak emango ditugu. Noski, pantailan zerbait margotzeko pixel baten koordenatuak behar dira. Beraz, gelaxka bati dagokion pixel koordenatuak lortzeko funtzio bihurtzaile bat beharko dugu.
- Board: taula bat irudikatzeko klasea.
- Tetris: jokoa kontrolatzeko beharko dugun klasea.

1. ARIKETA:

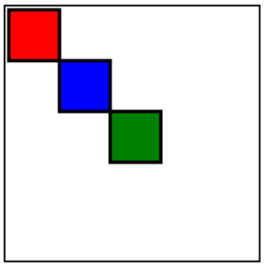
Sortu Block izeneko klasea, Rectangle klasetik heredatzen duena. Bere atributuak x eta y izan behar dira (gelaxka edo bloke baten posizioa adierazteko). Gogoratu (0,0) goi-ezkerreko erpina dela eta beraz, (9,19) behe-eskubikoa.

Zure kodea ondo programatuz gero (Point, Rectangle eta Block klaseak ondo badituzu), honako kodeak:

```
var block1 = new Block(new Point(0,0), 'red'),
block2 = new Block(new Point(1,1), 'blue'),
block3 = new Block(new Point(2,2), 'green');

block1.draw();
block2.draw();
block3.draw();
```

pantailan horrelako emaitza eman beharko luke (5x5 taula txiki batean):



Tetrominoak

Pieza bat irudikatzeko klase bat sortu, Shape izenarekin. Ikusi dugun bezala, pieza batek bloke zerrenda bat gordeko du. Baita draw() izeneko metodo bat, pieza pantailan margotzeko.

Behin klasea inplementatu ondoren, zure kodeari beste klase hau (I_Shape) gehituiozu. Klase honek Shape izeneko klasetik heredatu behar du.

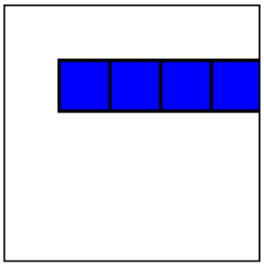
```
function I_Shape(center) {
    var coords = [new Point(center.x - 2, center.y),
            new Point(center.x - 1, center.y),
            new Point(center.x , center.y),
            new Point(center.x + 1, center.y)];

    Shape.prototype.init.call(this, coords, "blue");
}
```

Center parametroa Point klaseko objektu bat da, piezaren bloke zentralaren posizioa gordetzen duena (piezak posizio honekiko biratuko du, baina oraingoz ez dugu honetan sakonduko). Zure kodeak I_Shape pieza ondo marrazten duela egiaztatu. Hemen duzu adibide bat:

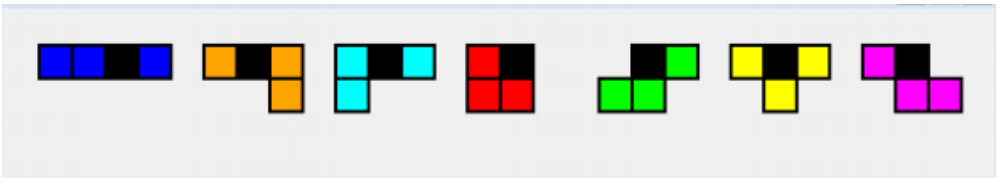
```
var shape = new I_Shape(new Point(3, 1));
shape.draw();
```

Pantailan ikusi beharko zenukeen emaitza honakoa da:



2. ARIKETA:

Jarraian gainontzeko piezak irudikatzeko klaseak sortu beharko dituzu. Pieza bakoitzak bloke berezi bat dauka beltzez markatuta, hori izango baita pieza bakoitzaren bloke zentrala (baina adi!, pantailan ez da beltzez ikusi behar, horrela margotu dugu hemen lagungarria izan daitekeelakoan)



Erabilitako koloreak: blue, orange, cyan, red, green, yellow eta magenta.

Hurrengo kodeak aurreko irudian ikusten diren pieza guztiak marrazten ditu:

// 7 tetromino pantailaratzeko zerrenda bat. Falta zaizkizun 6 klase inplmementatu beharko dituzu.

```
var tetrominoak = [I_Shape, J_Shape, L_Shape, O_Shape, S_Shape, T_Shape, Z_Shape];
x = 3;
tetrominoak.forEach( function ( tetromino ) {
                    shape = new tetromino(new Point(x, 1));
                    shape.draw();
                    x += 4;
                  });
```

Pieza berri bat ausaz sortu

Jokoaren nukleoa osatzen duten tetrominoak jada badakigu taularen barnean -canvas batean- margotzen. Jarraian jokoaren logika inplementatuko dugu.

3. ARIKETA:

Hasteko Tetris klaseko create_new_shape metodoa programatuko dugu. Metodo horrek ausaz sortu behar du pieza bat. Kontuan izan SHAPES atributuan jada baditugula pieza guztien izenak.

Sortu duzun piezari *current_shape* deituko diogu (uneko pieza). Geroxeago pieza hori izango da mugi edo biratu beharko duguna (erabiltzaileak teklatuarekin kontrolatuko duena). Bukatzeko, Tetris.init metodoa aldatu (eta horrekiko mendekotasunak dituztenak) uneko pieza taulan bistaratzeko.

Teklatuak sortzen dituen gertaerak kontrolatu

Piezak pantailatik mugitu ahal izateko, teklatuak sortzen dituen gertaerak kudeatu behar dira. Horretarako, Tetris.init metodoan teklatu kudeatzaile bat sortu dugu, (4. ariketaren txantiloian) evenListener erabiliz:

```
document.addEventListener('keydown', this.key_pressed.bind(this), false);
```

Kudeatzaile horrek key_pressed metodoari deituko dio erabiltzaileak tekla bat sakatzen duen bakoitzean:

```
Tetris.prototype.key_pressed = function(e) {

 var key = e.keyCode ? e.keyCode : e.which;

      // key aldagaian erabiltzaileak sakatu duen teklaren ASCII kodea
      // gordeko da. Zein da ezkerrera, eskubira, behera edo biratzeko
      // dagokion key kodea ?
}
```

Piezak mugitzen

Fijatu zaitez *Shape*, *Block* eta *Rectangle* klaseetan dauden *move* metodoak jada programatuta daudela, baita *Rectangle.erase* metodoa ere. Azken metodo horrek taulatik bloke bat ezabatzen du (uneko posizioa ezabatu eta blokea posizio berrian margotzen da, mugimendua simulatuz).

4. ARIKETA:

Teklatuko gertaerak kudeatu ditugu aurreko pausuan. Orain Tetris.key_pressed eta Tetris.do_move metodoak alda itzazu erabiltzaileak ⇦, ⇨, edo ⇩ sakatzen duenean pieza norabide horretan mugi dadin. Aztertu Tetris klasearen DIRECTION atributua. Bertan String motako gakoak dituen array bat aurkituko duzu. Gako bakoitzak norabide bat zehazten du eta balio gisa norabide horretan mugitzerakoan zenbat gelaxka X ardatzan eta zenbat Y ardatzan mugitu behar den zehazten da. Adibidez, Tetris.DIRECTION['Left'] gakoan, (-1, 0) balioa dugu. Horrek esan nahi du pieza bat ezkerrera mugitzeko, x = x + -1 eta y = y + 0 egin behar dela.

Taularen muga barruan mugitzen

Ikusi duzun bezala, uneko pieza taularen ezkerreko, eskubiko edo beheko mugatik at joan daiteke. Hau saihestu nahi dugu. Horretarako metodo batzuk aldatuko ditugu:

Board.can_move : orain arte beti bueltatzen zuen true balioa. Orain parametro gisa pasatzen zaion posizioa taularen muga barruan geratzen dela egiaztatzen du eta horren arabera true edo false itzultzen du.

Block.can_move: parametro gisa (dx,dy) desplazamendu bat jasotzen du. Uneko blokearen posizioa gehi (dx,dy) desplazamendua taularen muga barruan geratzen den edo ez kalkulatzen du. Horretarako aurreko Board.can_move metodoari deitzen dio. Posizio berrira mugitzea posible bada, true bueltatzen du. Bestela, false.

Shape.can_move: uneko piezaren bloke guztiak (dx,dy) gelaxka mugitzen baditugu taularen barruan geratzen diren egiaztatzen du. Horretarako Block.can_move erabiltzen du.

5. ARIKETA

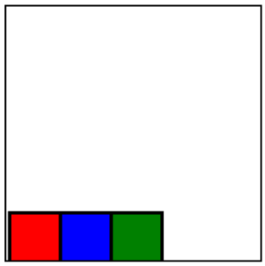
Ez digu denborarik eman Board klaseko can_move metodoa programatzeko. Laguntzerik bai?

Taulan pieza berri bat txertatu

Aurreko ariketa probatu baduzu ikusiko zenukeen pieza batek taularen beheko aldea ikutu ondoren ez dela bertan margotuta geratzen. Izan ere, nola antzeman pieza batek taularen hondoa ikutu duela? Nola gehitu pieza bat taulari? Zein izan beharko litzateke datu-egitura egoki bat taulari gehitu zaizkion piezak gordetzeko?

Taularen egoera eta bertan geratu diren piezak kudeatzeko hash motako datu egitura bat (hiztegi hitzarekin ere ezagutzen dena) erabiliko dugu. Hash horri grid izeneko taularen (Board klaseko) atributu batean gordeko dugu. Grid horren gakoak “x,y” posizioak izango dira. Eta “x,y” gako baten balioa bloke bat izango da. Adibidez, demagun horrelako 5x5 gelaxkako taula bat dugula. Bertan, 3 bloke geratu dira taularen beheko aldean. Orduan, gure grid datu-egitura horrela geratuko litzateke:

Gakoa	Balioa
“0,4”	bloke gorria objektua
“1,4”	bloke urdina objektua
“2,3”	bloke berdea objektua



6. ARIKETA

Board.add_shape metodoa programatu. Metodo honek pieza bat jasotzen du parametro gisa eta pieza horren bloke guztiak grid aldagaian gordetzen ditu (goiko aldean komentatu den bezala).

Tetris.do_move metodoa aldatu. Bertan mugimendu bat ea egitea posible den edo ez egiaztatzen dugu. Mugimendua ezin denean egin egiaztatu zergatik izan den. Alegia, erabiltzaileak behera (“Down”) mugitu nahi zuelako bada, orduan:

- uneko pieza (current_shape) taulan gehitu (grid aldagaian sartu).
- sortu beste pieza berri bat eta current_shape gisa gorde
- pieza berria taulan margotu

Piezen arteko talkak

Aurreko ariketa burutu ondoren jabetu al zara piezen arteko talkak ez ditugula kudeatzen? Hau da, piezak taularen beheko aldera jaisten dira baina tartean beste pieza bat aurkituz gero, teilakatu egiten dira. Hori ekidin nahi dugu, noski. Nola?

7. ARIKETA

Board.can_move metodoa aldatu uneko posizioan beste pieza bat dagoen egiaztatzeko.
True itzuli uneko posizioan gelaxka huts bat badago. False beste edozein kasutan.
Argibidea: ***in*** eragilea grid hiztegian erabili dezakezu “x,y” posizioan bloke bat dagoen jakiteko.

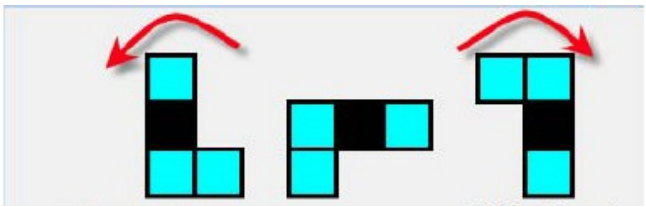
Piezak biratu

Piezak mugitzea ez da hain zaila izan, ez da? Pieza bat biratzea ere erraza da behin hori egiteko formula ezagutu ondoren. Honako formulak bloke bat beste batekiko 90 gradu biratzen du:

x = center.x -dir*center.y + dir*block.y
y = center.y + dir*center.x -dir*block.x

Formula horretan:

- dir aldagaiak norabidea gordetzen du. dir = 1 bada, erloju-orratzen norabidean biratuko du. dir = -1 bada, erloju-orratzen kontrako norabidean.
- center: biraketea egiteko erreferentzia gisa hartzen den blokea. Adibidez, hurrengo irudian center bloke beltza izango litzateke. L_Shape pieza biratzeko erreferentzia gisa hartzen den blokea, hain zuzen ere.
- block: biratu nahi den blokea



Jatorrizko Tetris jokoan xehetasun bat dago piezen biraketan. J, L eta T piezek erloju-orratzen norabidean biratzen dute. I, S eta Z piezez, ordea, oszilatu (kulunkatu) egiten dute, alegia, lehenengo erloju-orratzen norabidean biratzen dute eta ondoren kontrako norabidean, patroi hau behin eta berriro errepikatuz). Bestalde O (lauki gorria) piezak ez du biratzen. Aurreko datu guztiak kontutan izanik, eta zure lana errazte aldera, Shape.can_rotate eta Shape.rotate jada programatu ditugu. Zure lana metodo horiek erabiltzea besterik ez da izango.

8. ARIKETA

Tetris.do_rotate metodoa programatu, uneko pieza biratzeko (biraketa posible bada). Gogoratu Shape.can_rotate eta Shape.rotate jada programatuta daudela!

Tetris.key_pressed metodoa aldatu, ↑ tekla sakatzerakoan uneko pieza rota dadin.

Piezak behera mugitzen

Pieza bat taularen beheko aldera jaisteko hiru modu daude. Teklatuan ⇩ sakatu, teklatuan zuriune-barra sakatu edo ezer sakatu gabe denbora pasatzen utzi (automatikoki jaitsiko da beheruntz, gelaxka bat segunduko). Lehenengo aukera jada programatu dugu baina beste biak falta zaizkigu.

9. ARIKETA

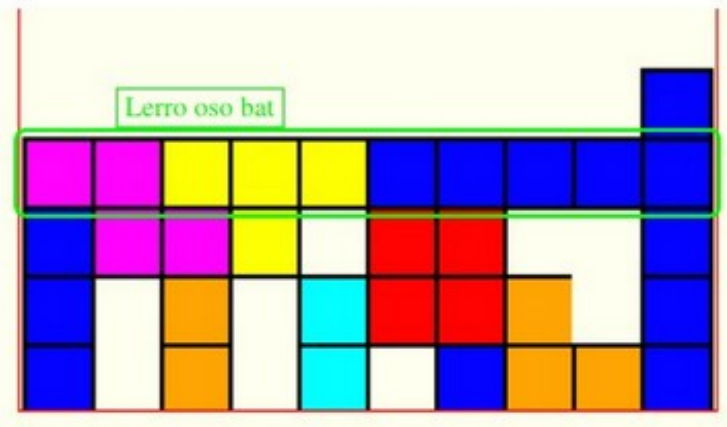
- Tetris.key_pressed metodoa eguneratu erabiltzaileak zuriune barra sakatzen badu uneko pieza berehala taularen beheko aldera jaitsi dadin.

- Erabiltzaileak ez badu ezer egiten (ez badu teklaren bat sakatzen) uneko pieza automatikoki behera joan behar da, gelaxka bat segunduko. Horretarako Tetris.animate_shape metodoa sortu. Metodo hori automatikoki exekutatu behar da segundoro, uneko pieza gelaxka bat

behera mugi dadin. Tetris.init metodoa aldatu beharko duzu, baita ere, hasierako deia burutzeko (animate_shape metodoari deitu hemendik).

Lerro osoak ezabatzen

Zure jokoa ia-ia bukatu duzu. Baina azken ikutua falta zaio: pieza bat taulari gehitzerakoan (grid aldagaian sartzerakoan), egiaztatu behar da ea lerro oso bat sortu den (lerro bat non gelaxka guztiak blokez osatuta dagoen). Kasu horretan lerro hori ezabatu egin behar da, alegia, lerroa sortzen duten blokeak grid-etik ezabatu egin behar dira. Baina hor ez da arazoa amaitzen. Ezabatu dugun lerroaren gainetik dauden bloke guztiak gelaxka bat behera mugituko dira. Eta noski, hori egiterakoan lerro berriak sor daitezke, ezabatu egin behar direnak, etab.



Ikastaro honetan lerro osoak ezabatzeko metodoak jada inplementatuta ematen zaizkizu. Zehazki, Board (taula) klasean:

- is_row_complete: true itzuliko du parametro gisa pasatzen zaion lerroko gelaxka guztiak okupatuta badaude.
- delete_row: lerro bat ezabatzen du.
- move_down_rows: parametro gisa pasatzen diogun lerro gainetik dauden lerro guztiak gelaxka bat behera mugitzen ditu.
- remove_complete_rows: lerroren bat osoa dagoen aztertzen du. Balego, ezabatu egiten du, gainetik dituen lerro guztiak gelaxka bat behera mugituz.

10. ARIKETA

Tetris.do_move metodoa aldatu egin beharko duzu. Orain, taulari pieza bat gehitzerakoan (grid aldagaian sartzerakoan), lerro osoren bat sortu den egiaztatu behar da eta horrela balitz, ezabatu. Metodo horretan ere, egokia litzateke aztertzea ea uneko pieza berria (0,0) desplazamenduarekin taulan margotu dezakegun. Alegia, ea pieza berria taularen goiko aldean jartzen dugunean, inolako desplazamendurik egin gabe [(0,0) = ez dago desplazamendurik] ea bertan margotu dezakegun. Ezin badugu --> GAME OVER. Kontuan izan Game Over egiterakoan komenigarria izango litzatekela, baita ere, piezak segundoro mugitzeko martxan dagoen erlojua etetea.

Zorionak! Honeraino heldu bazara, jokoaren programazioarekin bukatu duzu :-) Gehiago nahi? Puntuazioa, soinu efektuak, musika, hurrengo piezaren aurrebista, ... Gauza asko gehitu daitezke. Eutsi erronkari!