LATVIAN IT CLUSTER

NACIONĀLAIS ATTĪSTĪBAS PLĀNS 2020

EIROPAS SAVIENĪBA
Eiropas Reģionālās attīstības fonds

IEGULDĪJUMS TAVĀ NĀKOTNĒ

# Datorspēļu veidošana

Mācību materiāls

# Datorspēļu veidošana

Projekta Nr. Nr.3.2.1.1/16/A/004 **“Informācijas tehnoloģiju nozares konkurētspējas un eksportspējas paaugstināšanas”** ietvaros

LATVIAN IT CLUSTER

NACIONĀLAIS ATTĪSTĪBAS PLĀNS 2020

EIROPAS SAVIENĪBA
Eiropas Reģionālās attīstības fonds

IEGULDĪJUMS TAVĀ NĀKOTNĒ

# Saturs

Ievads ..... 4
Programmēšana. Programmēšanas valodas ..... 5
Datorspēles jēdziens ..... 7
Datorspēles izstrādes process ..... 8
Spēles koncepcijas izstrāde ..... 8
Datorspēles izstrāde ..... 8
Datorspēles izplatīšana ..... 9
Datorspēļu izstrādes programma Construct 2 ..... 9
Jauna projekta saglabāšana ..... 10
Vienkāršas spēles izveide ..... 12
Autortiesības ..... 39
Izmantoto informācijas resursu saraksts ..... 41

# Ievads

Mācību materiāls “Datorspēļu veidošana” ikvienam interesentam sniedz priekšstatu par datorspēļu izstrādes posmiem, būtiskāko katrā no tiem, kā arī nelielu ieskatu datorspēļu izstrādes vēsturē un nākotnes attīstības tendencēs.

Datorspēļu izstrāde principā nav iedomājama bez programmēšanas, tāpēc kopējam priekšstatam materiālā iekļauta arī informācija par šobrīd populārākajām programmēšanas valodām, kā arī skaidroti ar programmēšanu saistīti jēdzieni.

Jāpiebilst, ka materiālā nav sniegtas zināšanas, kā veidot datorspēli, izmantojot konkrētas programmēšanas valodas jeb programmējot, bet gan ir apskatīta populāra spēļu izveides programma Construct 2, ar kuras palīdzību tie, kas vēl neprot programmēt, var izveidot interesantas datorspēles.

Mācību materiāls noteikti ir piemērots iesācējiem, kas nav darbojušies ar programmu Construct 2 un vēlas apgūt tās piedāvātās iespējas.

Programmas Construct 2 iespējas ir visai plašas, tāpēc tās padziļinātai apguvei kā papildus materiālu noteikti ieteicams izmantot projekta “Generation 0101” ietvaros izveidoto studentu grāmatu “Programmēšanas pamati”, kā arī iepazīties ar resursiem, uz kuriem saites iekļautas šī materiāla beigās.

# Programmēšana. Programmēšanas valodas

**Datorprogrammēšana** jeb programmēšana ir datorprogrammas izveides process.

Tas ietver datorprogrammas pirmkoda projektēšanu, rakstīšanu, atkļūdošanu, testēšanu un uzturēšanu. Šis pirmkods tiek rakstīts kādā programmēšanas valodā. Programmēšanas nolūks ir izveidot lietojamu programmu, kas atbilst iecerētajai darbībai un parametriem. Šis process bieži prasa īpašas zināšanas daudzās dažādās jomās, piemēram, programmas pielietošanas sfērā, specializētu algoritmu pārzināšanu un formālo loģiku.[1]

**Programmēšanas valoda** - formāla valoda, kas kalpo datorprogrammu aprakstam.[2]

Mūsdienās programmēšanas valodu zināšana un prasmes programmēšanā paver plašas iespējas darba tirgū. Saskaņā ar Eiropas Komisijas datiem IKT nozare strauji attīstās un ik gadu rada aptuveni 120000 jaunu darba vietu. Prognozes liecina, ka laika posmā līdz 2020. gadam Eiropā varētu pietrūkt pat līdz 900000 kvalificētiem IKT darbiniekiem.

Ik dienas medijos parādās informācija par izveidotajiem topiem visdažādākajās jomās: mūzikā, mākslā, sportā, auto rūpniecībā, modes industrijā un daudzās citās. Izņēmums nav arī programmēšanas valodas. Iepazīsimies ar vietnē www.techworm.net publicēto programmēšanas valodu topu 2017. gadā un pievērsīsim uzmanību, ka daudzas no šīm valodām tiek izmantotas arī spēļu programmēšanā.

1. - **Java**

**Java** ir programmēšanas valoda, kas izstrādāta pagājušā gadsimta 90. gados. Kā šīs valodas priekšrocība tiek uzskatīta spēja darboties uz dažādu platformu operētājsistēmām, platformām un ierīcēm. To izmanto, piemēram, Android lietotņu, video spēļu un daudzu citu digitālo rīku programmēšanā.

2. - **C**

**C**, kas ir valodas C++ priekštece, radīta pagājušā gadsimta 70. gados un šobrīd ir vecākā no pieprasītajām programmēšanas valodām. Mūsdienās ar tās palīdzību raksta operētājsistēmu kodus, kā arī kodus dažādām citām sistēmām. Profesionāļi, kas prot programmēt valodā **C**, darba tirgū tiek ļoti augstu novērtēti.

3. - **C++**

---

[1] https://lv.wikipedia.org/wiki/Programm%C4%93%C5%A1ana
[2] https://lv.wikipedia.org/wiki/Programm%C4%93%C5%A1anas_valoda

Jau vairāk nekā desmit gadus **C++** populārāko programmēšanas valodu topā ir atrodama starp trešo un piekto vietu.

Tiek uzskatīts, ka **C++** salīdzinājumā ar C ir vienkāršāka. Ar šīs valodas palīdzību raksta, sākot no pārlūkprogrammām līdz pat datorspēlēm. Pazīstamās operētājsistēmā Windows programmēšanā lielā mērā arī ir izmantota valoda C++.

4. – **C#**

**C-Sharp** izstrādājusi un ikdienā izmanto kompānija "Microsoft" platformai ".NET". Ar šīs valodas palīdzību visbiežāk arī tiek veidotas Windows operētājsistēmas.

5. - **Python**

**Python** ir plaši izmantota augsta līmeņa programmēšanas valoda, turklāt vienkārša. Tās ietvaru (framework) Django izmanto ļoti plaši, sākot no NASA un Google līdz pat Instagram un Pinterest. Tiek uzskatīts, ka zināšanas Python programmētāja karjeras izaugsmē ir būtisks faktors.

6. - **PHP**

**PHP** (hiperteksta Pre-procesors) ir servera puses programmēšanas valoda, ko var izmantot, lai izveidotu interneta lapas, kas rakstītas HTML. Tā tiek izmantota vairāk nekā 80 % pasaulē izstrādāto mājas lapu, tajā skaitā Facebook, Wikipedia, Tumblr un WordPress. PHP ir ļoti populāra jauno programmētāju vidū.

7. - **JavaScript**

**JavaScript** mūsdienās ir viena no izplatītākajām programmēšanas valodām. **JavaScript** tiek izmantota, piemēram, lai izveidotu interaktīvas un animētas mājas lapas, kā arī spēles internetā. JavaScript ir saderīga ar visām pārlūkprogrammām, un tā izmanta vairāk nekā 90 % visu tīmekļa lapu.

8. - **Visual Basic .NET**

**Visual Basic .NET** ir izstrādājis uzņēmums Microsoft, un tā ir valodas VisualBasic pēctece.

9. - **Delphi/Object Pascal**

**Delphi** ir programmēšanas valodas Turbo Pascal pēctece. Pirms pāris gadiem Delphi popularitāte samazinājās, un tā programmu topā ieņēma vien 20. vietu, savukārt 2017. gada februārī tā jau atkal bija atrodama pirmajā desmitā.

10. - **Perl**

**Perl** sākotnēji tika izstrādāta 1987. gadā kā universālu UNIX skriptu valoda, lai ziņojuma apstrādi padarītu vieglāku. Mūsdienās **Perl** izmanto sistēmas administrēšanas, tīkla plānošanas, finanšu, bioinformātikas un citu lietojumprogrammu izstrādē.

# Datorspēles jēdziens

Kas ir datorspēle?

Portāla www.letonika.lv Terminu un svešvārdu skaidrojošajā vārdnīcā pieejama šāda definīcija:[3] **Datorspēle** - izplatīts izklaidēšanās veids, izmantojot datoru interaktīvā režīmā. Datorspēles aptver plašu interešu loku - sākot no alfabēta mācīšanas maziem bērniem līdz šaham, kara spēlēm, kā arī dažādu sadzīves un ekonomikas problēmu risināšanas algoritmu meklēšanai.

Mūsdienās jēdzienu "datorspēle" dzirdam ļoti bieži, un pasaulē ik dienas tiek radīts liels daudzums dažāda satura datorspēļu. Rodas jautājums, cik sena ir datorspēļu rašanās vēsture un kad tika izveidotas pirmās datospēles?

Iepazīstoties ar dažādiem informācijas avotiem jāsecina, ka tajos publicētā informācija atšķiras, piemēram, vietnē Inside Gaming[4] minēts, ka pirmā spēle, kura simulēja raķetes tuvošanos mērķim, tika radīta 1947. gadā. 1952. gadā A.S. Duglass (A.S. Douglas) radīja krustiņu – nullīšu spēli OXO. Pastāv arī uzskats, ka pirmā datorspēle tika radīta 1958. gadā. Spēles autors Viljams Higinbotams (William Higinbotham) radīja ļoti vienkāršu tenisa simulatoru.

Savukārt, kā liecina 2007. gadā žurnālā "Sakaru pasaule" publicētā informācija, tad pirmā datorspēle tika izveidota 1962. gadā. Programmētāja Stīva Rasela (Steve Russell) vadītā MIT (Massachusetts Institute of Technology) programmētāju grupa, iedvesmojoties no E. Smita fantastikas grāmatām, izveidoja pirmo datoru videospēli ar nosaukumu "Kosmosa kari" (SpaceWar).

Spēli "SpaceWar" varēja spēlēt divi spēlētāji, kas vadīja bruņotus kosmosa kuģus, kuri mēģina viens otru sašaut, manevrējot zvaigžņu gravitācijas laukos. Katram kuģim bija ierobežots lādiņu un degvielas daudzums. Pirmajās versijās zvaigznes izplatījumā ģenerēja nejaušo skaitļu ģenerators, taču vēlāk spēle tika papildināta ar kodu, kas atainoja pēc reālām zvaigžņu kartēm veidotu debesjumu. Tas lēnām rotēja, vienlaikus ļaujot aplūkot 45 procentus naktī redzamo zvaigžņu. Spēlētāju uzdevums bija satriekt pretinieka kosmisko kuģi, neciešot apšaudē un nenokļūstot bīstami tuvu zvaigznēm. Šis spēles veids desmit gadus vēlāk tika nosaukts par arkādi, iespaidojoties no gaiteņa ar arkveida pārsegumu, kurā tradicionāli tika izvietoti spēļu automāti.

Divus gadus pēc spēles pabeigšanas studentiem sāka šķist, ka datu ievade no klaviatūras neatbilst spēles ideoloģijai. Datorspēles projekta dalībnieki Alans Kotaks un Roberts Sanders, izmantojot Stīva Rasela ideju, izveidoja jaunu vadības ierīci - pirmo džoistiku. Ar tā kreiso sviru varēja stūrēt kuģus, bet ar labo - regulēt pārvietošanās ātrumu. Džoistikam bija arī poga raķešu palaišanai. Pēc neilga laika SpaceWar oficiāli tika atzīta par pirmo videospēli.

---

[3] http://www.letonika.lv/groups/default.aspx?r=1107&q=datorsp%C4%93le&id=2072173&g=1
[4] http://old.pvg.edu.lv/datori/konkursi/2011_web/Kristaps_Znotins/vesture.htm

Datorspēļu izstrādes industrija ir attīstījusies ļoti ātri. Mūsdienās kaut arī sakām "datorspēle", tomēr šim jēdzienam ir daudz plašāka nozīme, jo mēs vairs nerunājam par spēles spēlēšanu tikai datorā, izmantojot džoistiku. Spēļu ierīces attīstījušās un varam izmantot gan virtuālās realitātes brilles, gan izliektus monitorus, daudzkanālu austiņas un citus risinājumus, kas spēlētājam nodrošina daudz lielāku klātbūtnes efektu spēļu pasaulē, kā arī piesaista daudz vairāk spēlētāju.

# Datorspēles izstrādes process

Datospēles izstrādes process, līdzīgi kā filmas izstrādes process, ietver vairākus svarīgus un savā starpā cieši saistītus posmus.

## Spēles koncepcijas izstrāde

Datorspēles izveidē pirmais posms ir **spēles koncepcijas izstrāde**, kura laikā tiek apspriesta spēles ideja un tās attīstība. Pati spēles ideja ir ļoti nozīmīgs faktors spēles izveides procesā, jo oriģināla, patērētāju pieprasījumam un datorspēļu attīstības tendencēm atbilstoša ideja noteikti ir svarīgs priekšnosacījums pozitīva rezultāta sasniegšanai.

Datorspēles izstrāde sākas ar tās projektēšanu, kad tiek veidots spēles dizaina dokuments, kurā apraksta spēli un tās funkcijas. Uzņēmumos, kas darbojas spēļu izstrādes jomā, projektēšanas posmā kopā darbojas speciālistu komanda, kas sastāv no programmētājiem, māksliniekiem, skaņu inženieriem un dizaineriem.

Bieži vien, runājot par spēles vai arī mājas lapas dizainu, tie, kas nav šīs jomas speciālisti, iedomājas tikai vizuālo noformējumu, taču tā tas nav – jēdzienam dizains ir daudz plašāka nozīme.

Spēles dizaina dokuments sastāv no trim daļām:

- Spēles koncepcijas apraksts;
- Spēles vizuālais noformējums;
- Spēles tehniskais dizains.

Kad spēles izsrādes komandai ir skaidrs sasniedzamais rezultāts, notiek nepieciešamo resursu plānošana, apzināšana. Tiek domāts, kādi rīki nepieciešami, kādi speciālisti, sadarbības partneri iesaistāmi, kā arī saskaņā ar veicamajiem darbiem tiek izveidots laika grafiks, kurā norāda, kādos termiņos ir jāpaveic konkrēti uzdevumi.

## Datorspēles izstrāde

Kad spēles izstrādes komanda ir gatava darbam – ir skaidra koncepcija, ir nepieciešamie resursi, kā arī izstrādāts laika grafiks, var sākties datorspēles izstrādes posms, kas atkarībā no produkta sarežģītības pakāpes var ilgt no viena līdz daudziem mēnešiem.

Uzņēmumi, kas darbojas datorspēļu izstrādes jomā, uzreiz neizstrādā gatavu produktu, šajā gadījumā spēli, bet gan tās prototipu[5] - paraugu vai modeli, kas tiek testēts, pārbaudīts un nepieciešamības gadījumā pilnveidots.

Testēšanas laikā ir svarīgi novērtēt spēles kvalitāti, sameklēt visas iespējamās kļūdas. Jāpiebilst, ka šajā posmā iesaistās ne tikai paši spēles veidotāji, kuri jau ir pazīstami ar produktu, bet tiek veidotas arī fokusa grupas, kas sastāv no neatkarīgiem ekspertiem. Šo papildus iesaistīto cilvēku uzdevums ir novērtēt spēli kopumā no dažādiem aspektiem, atsevišķus tās elementus, sniegt informāciju veidotājiem par pozitīvo, kā arī par trūkumiem, ko viņi kā datorspēles izmēģinātāji ir pamanījuši.

Testēšanas process var tikt atkārtots vairākkārt, līdz tiek uzskatīts, ka vairāk kļūdu un nepilnību nav un var tikt izstrādāts datorspēles galīgais variants.

## Datorspēles izplatīšana

Kad datorspēle gatava, to ir nepieciešams eksportēt un publicēt, lai tā būtu pieejama ikvienam interesentam. Jāpiebilst, ka mūsdienās dominē datorspēļu izplatīšana tiešsaistē.

Lai atvieglotu spēļu pārdošanu, dažādi spēļu izstrādes uzņēmumi ir radījuši savas tiešsaistes platformas, piemēram, viena no tām ir Games Marketplace.

# Datorspēļu izstrādes programma Construct 2

Construct 2 ir programma, kas paredzēta 2D spēļu veidošanai. Tā ļauj ikvienam veidot spēles bez zināšanām un prasmēm programmēšanā.

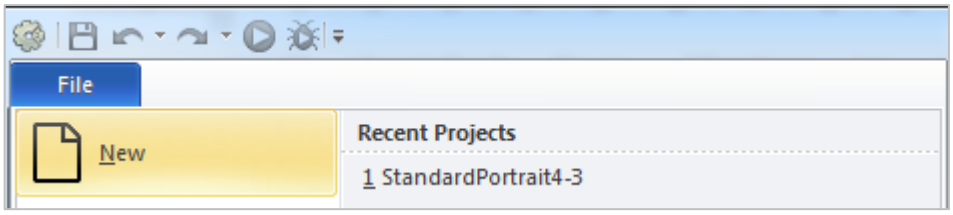
Construct 2 redaktors ir piemērots lietošanai tikai operētājsistēmā Windows, taču radītās spēles var spēlēt jebkurā vidē.

The Construct 2 start page

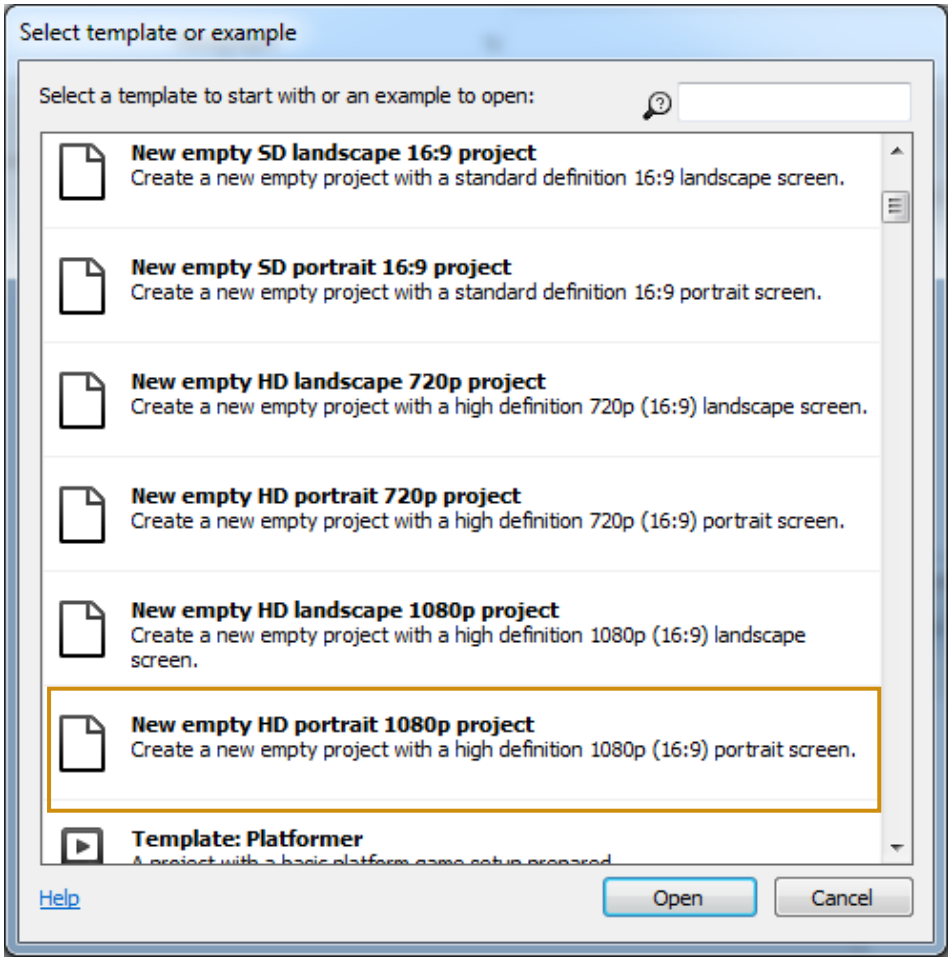
[5] http://termini.lza.lv/term.php?term=prototips&list=prototips&lang=LV

## Jauna projekta saglabāšana

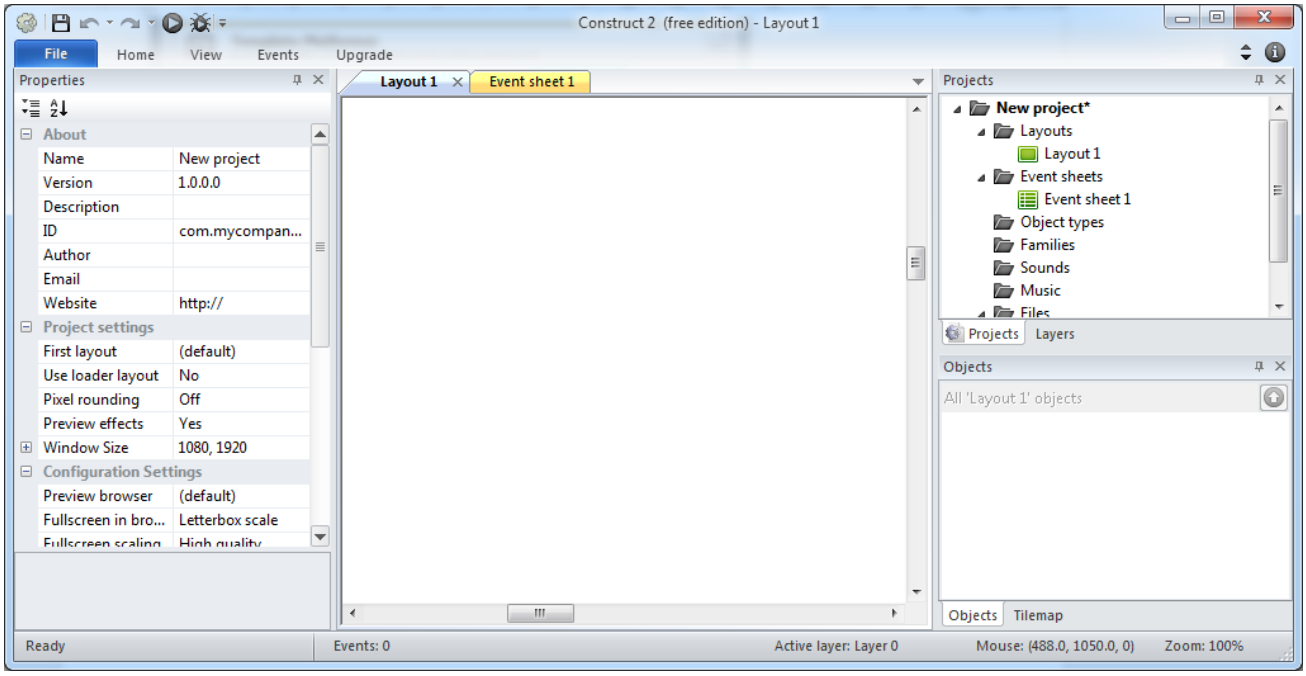
Lai programmā **Construct 2** uzsāktu jaunu projektu, klikšķina uz **File**, pēc tam uz **New**.



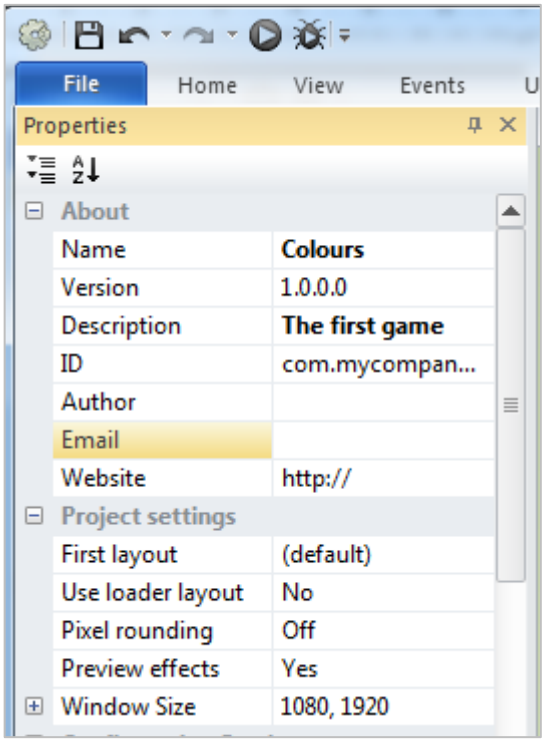

Kā redzams, tiek piedāvātas daudzas iespējas:




Šoreiz izvēlēsimies **New empty HD portrait 1080p project** – spēles loga attiecība būs 16:9, un tā novietojums - Portretorientēts. Noklikšķinot uz pogas **Open**, atvērsies logs, kurā ievada datus par projektu: tā nosaukumu, versiju, aprakstu utt.




Jāpiebilst, ka identifikatoru vēlāk izmanto, ja vēlas spēli publicēt **Google Play** vai **App Store**.



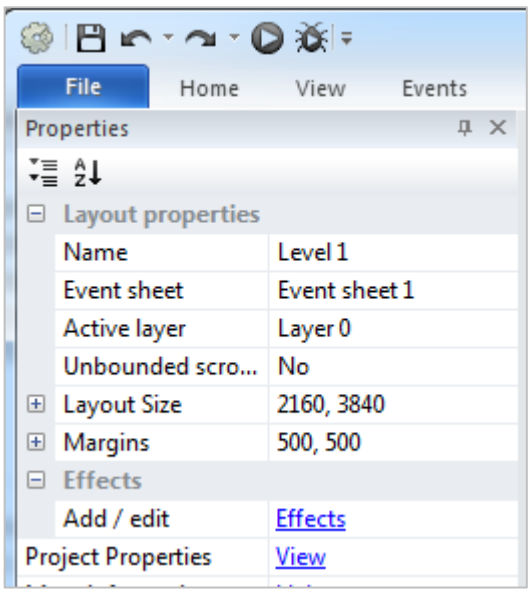
Lai izveidoto projektu saglabātu vienā failā, izvēlas **File/Save As Single File**. Ja plānots darboties komandā, tad izdevīgāk būtu saglabāt mapē **File/Save As Project**, kas vēlāk ļaus izmantot versiju kontroles sistēmas.

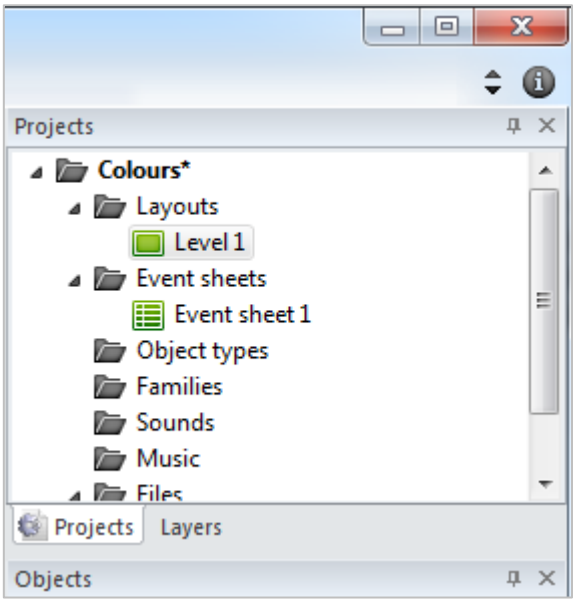
## Vienkāršas spēles izveide

Pēc projekta saglabāšanas ir pieejamas divas cilnes: **Layout 1** un **Event sheet 1**.

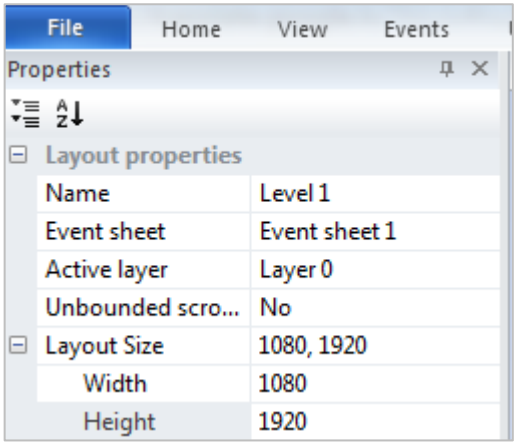
Pirmajā no tām būs apskatāms spēles ekrāna izkārtojums tā, kā to redzēs spēlētājs, otrajā tiks atspoguļoti notikumi. Šajā gadījumā **Layout 1** tiks izmantots spēles pirmajam līmenim. Cilnes nosaukuma maiņai uzklikšķina uz tās sadaļā **Projects**, pēc tam sadaļā **Properties** ieraksta jauno nosaukumu.

Kā redzams attēlā, **Layout 1** tika nomainīts uz **Level 1**.

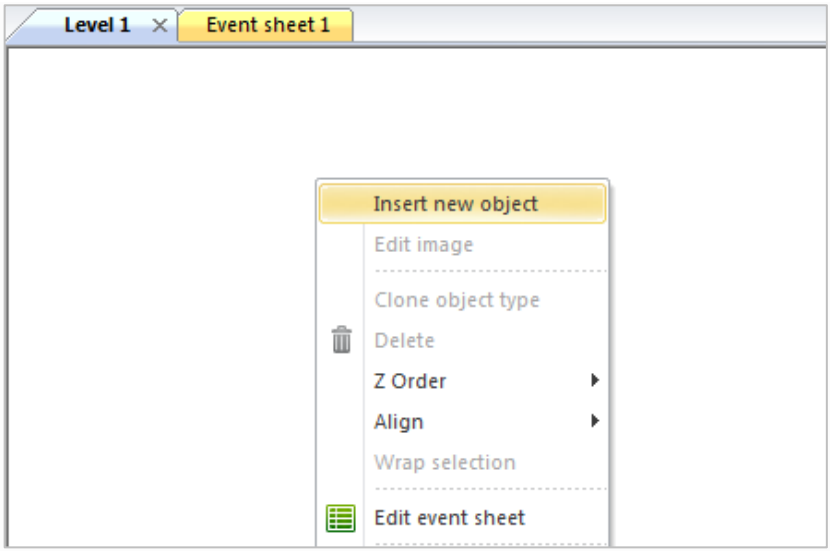


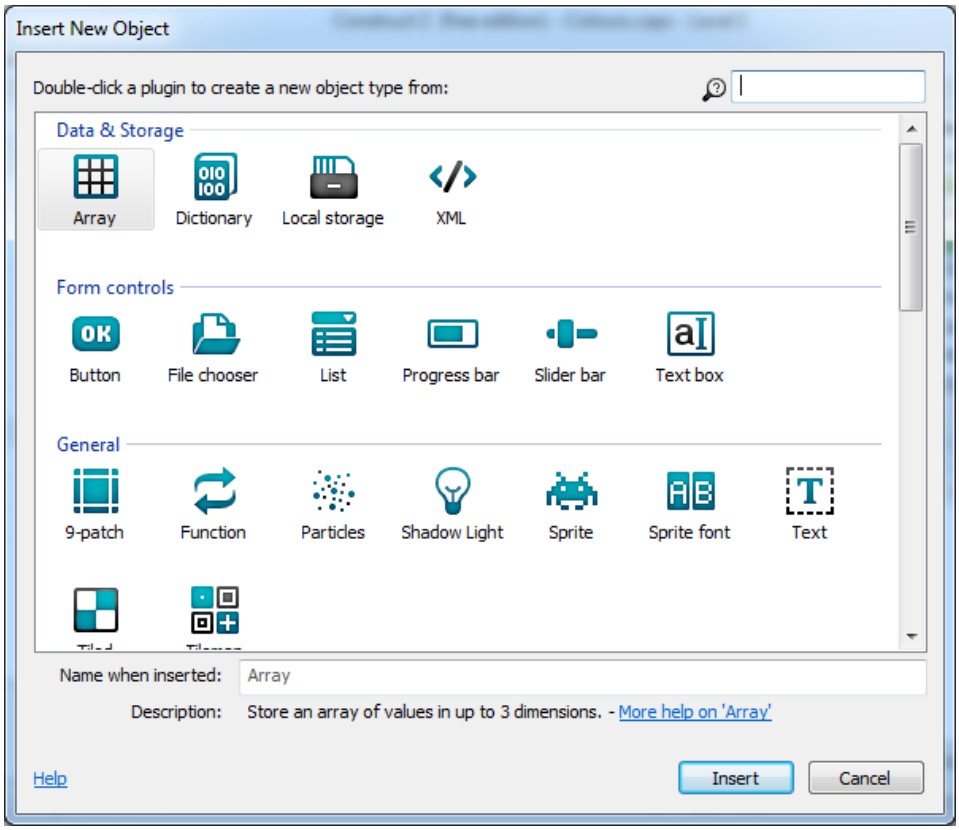

Sadaļā **Layout Size** nomaina arī maketa izmēru uz tādu, kādu sākumā norādīja loga izmēru – **1080x1920**.



Lai spēlei pievienotu objektus, baltajā laukumā veic klikšķi ar peles labo taustiņu un izvēlas **Insert new object**.

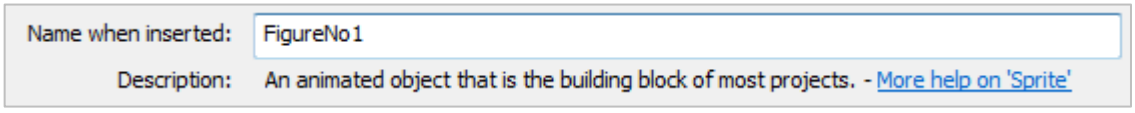



Atveras logs **Insert new object**, kurā tiek piedāvātas dažādas iespējas:



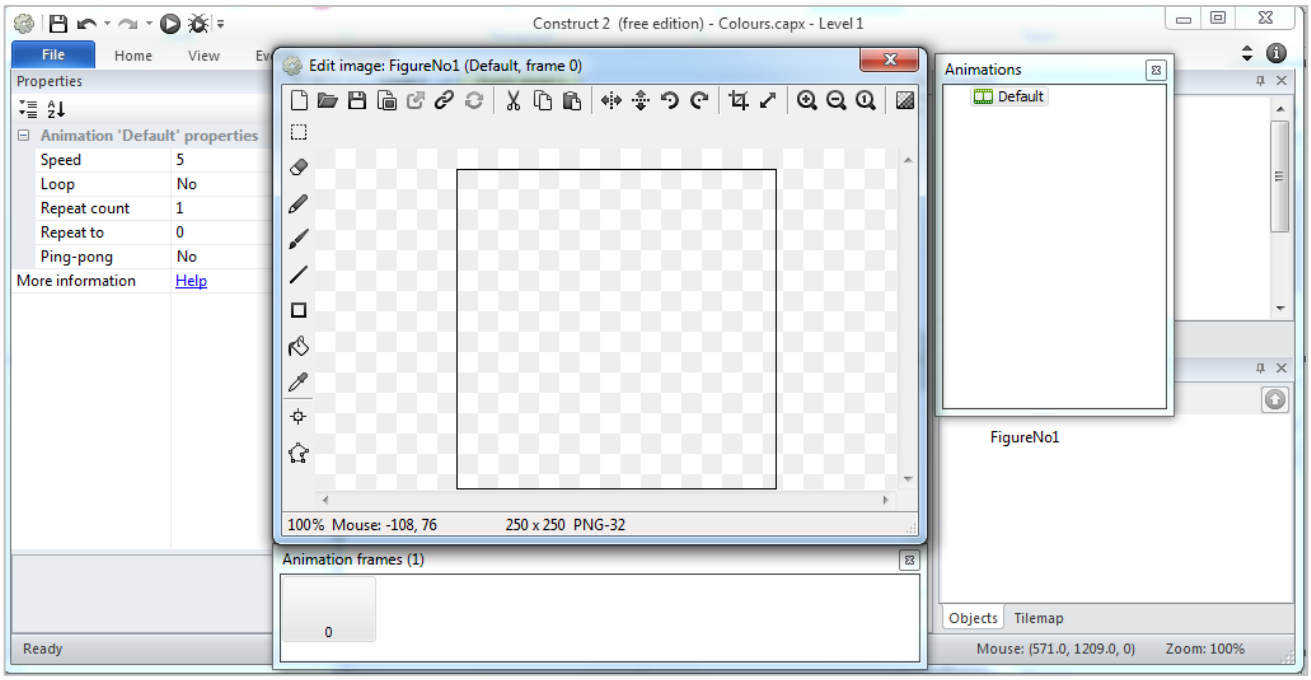
Lai pievienotu zīmētus objektus, izvēlas sadaļu **Sprite** (objekts – attēls).

Logā **Name when inserted** ievada jaunā objekta nosaukumu un noklikšķina uz **Insert**. Jāpiebilst, ka programma neļauj ievadīt nosaukumu, kurā starp vārdiem ir atstarpes.



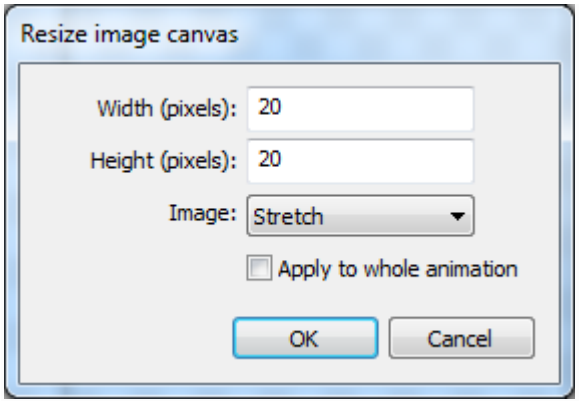

Ar peles kreiso taustiņu veic klikšķi laukumā, kur ievietos jauno objektu. Atveras vairāki logi.

Logs **Edit image**, kurā zīmē jauno objektu, kā arī **Animations** un **Animation frames**, kuros darbojas ar animācijām.

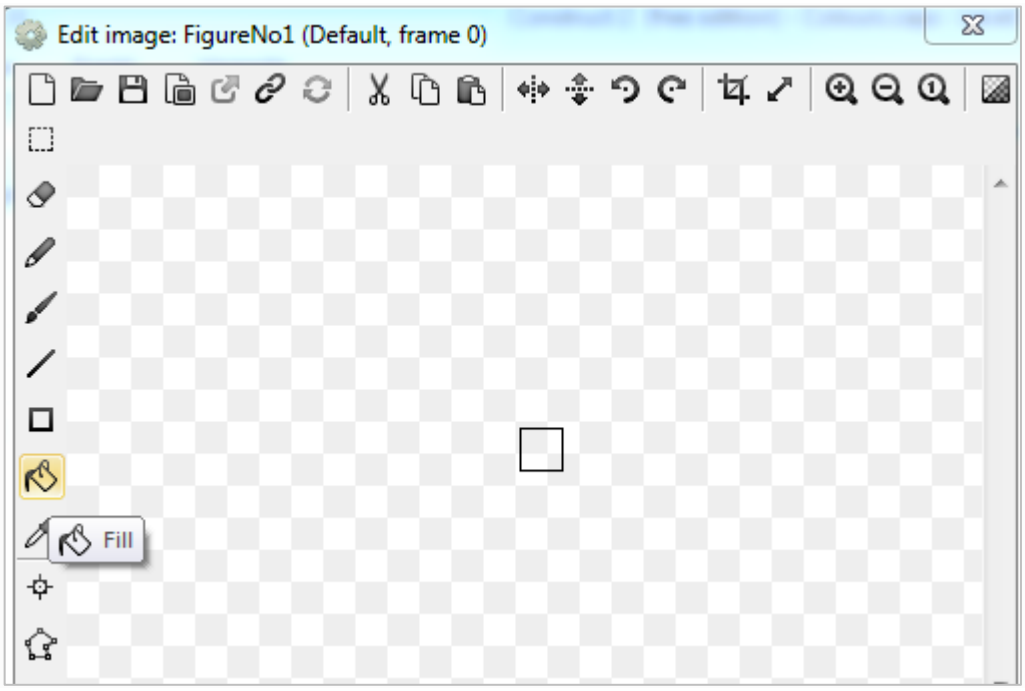



Ja nepieciešams, var mainīt loga objekta izmērus. Noklikšķina uz pogas **Resize.**

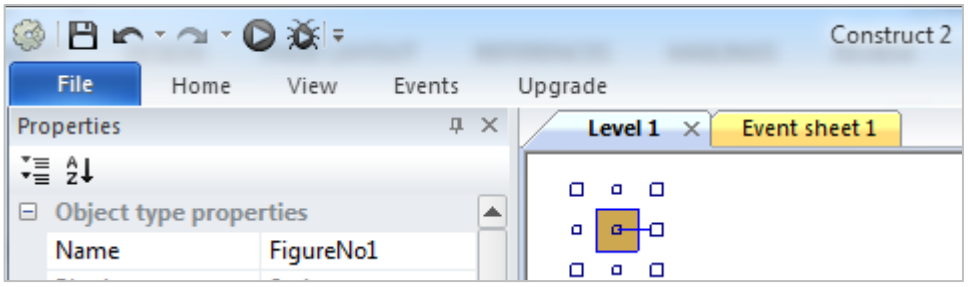
Atveras logs, kurā norāda nepieciešamos izmērus. Kā redzams attēlā, jaunais izmērs būs 20x20 pikseļi.




Lai zīmēto objektu aizpildītu ar krāsu, noklikšķina uz pogas **Fill**.



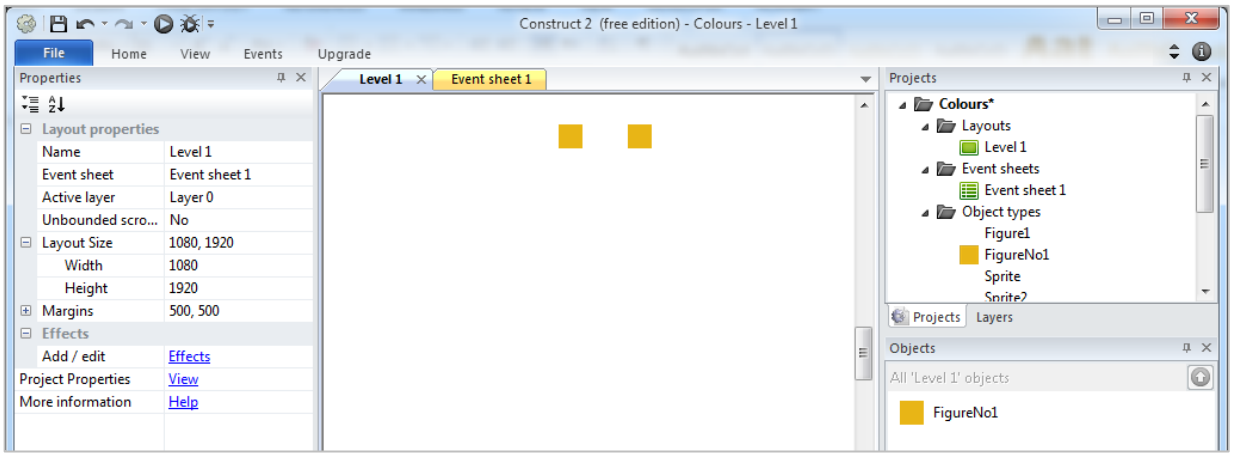

Kad objekts nokrāsots izvēlētajā krāsā, logu **Edit image** var aizvērt. Uzklikškinot uz objekta, var mainīt tā izmērus, kā arī pārvietot.



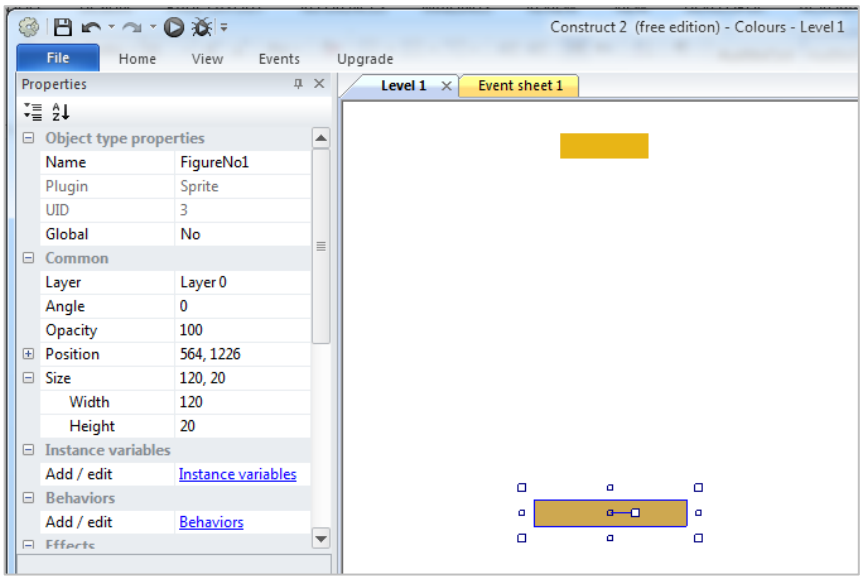
Pēc objekta izveidošanas tas redzams arī sadaļā **Projects**.

Lai zīmējumam pievienotu vēl vienu tieši tādu pašu objektu, izmantojami vairāki paņēmieni: viens no tiem – objektu no sadaļas **Projects** ievelk maketā **Level1**, var izmantot arī pazīstamās komandas **Copy** un **Paste**.

Kā redzams attēlā, iegūti divi vienādi kvadrāti.

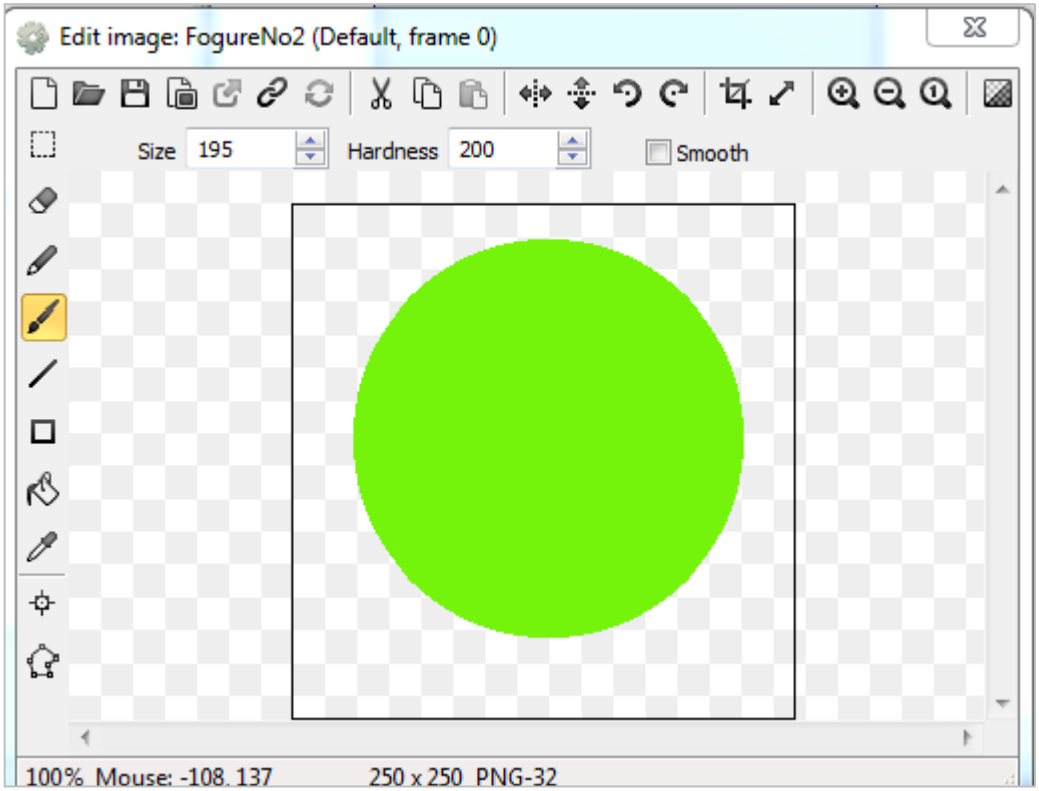



Uzklikšķinot uz katra no objektiem, var mainīt tā izmēru un novietojumu. Piemēram, viena no taisnstūriem jaunais izmērs ir 70x20, savukārt otrs tiek pārvietots zemāk, tā izmērs ir 120x20.

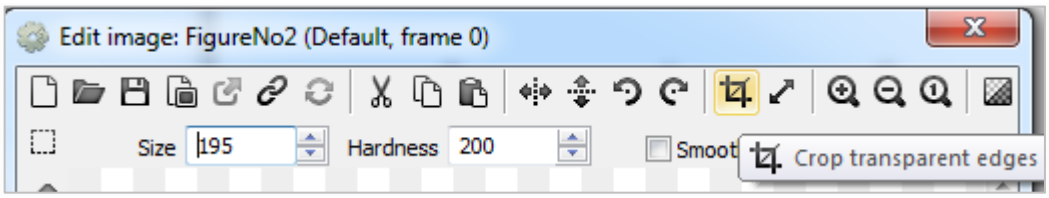


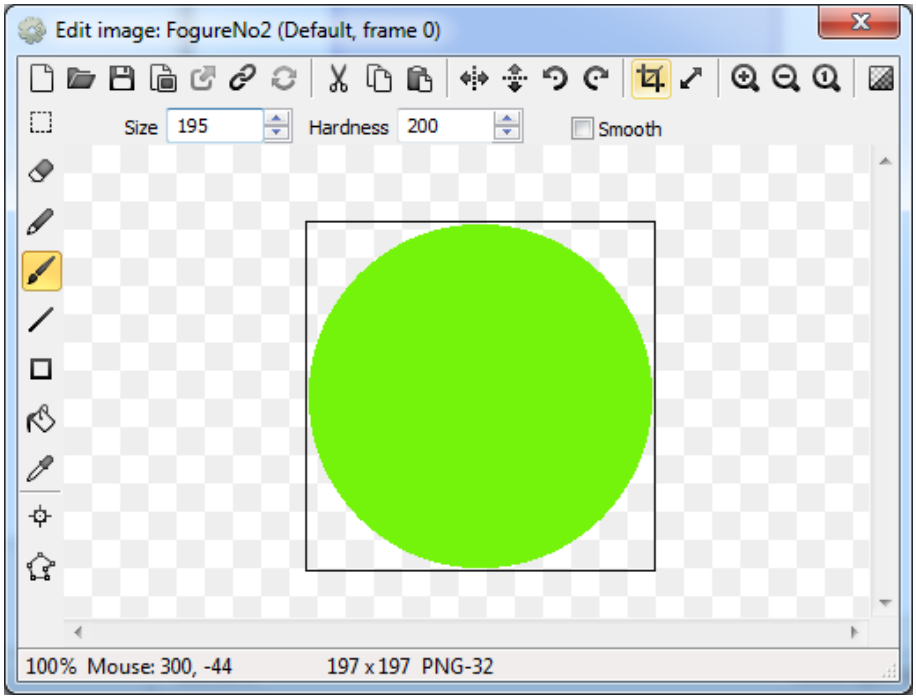
Nākošajā solī figūru skaitu papildina ar divām jaunām – zaļu un oranžu riņķi. Baltajā laukumā veic klikšķi ar peles labo taustiņu un izvēlas **Insert new object**. Pēc tam atkal izvēlas sadaļu **Sprite**. Lai uzzīmētu zaļo riņķi, izvēlas instrumentu **Brush**. Jauno objektu nosauc **FigureNo2**.

Lai iegūtu riņķi, jārīkojas sekojoši: izvēlas zaļo krāsu un otas izmēru mazāku nekā laukuma izmērs, piemēram, 195. Tiek piedāvāts izvēlēties arī otas cietību. Šajā gadījumā ir izvēlēts maksimāli iespējamais – 200. Jo lielāka otas cietība, jo noteiktāka riņķa kontūra. Jo šis parametrs mazāks, jo kontūra vairāk izplūdusi. Peles rādītāju novieto attēla vidū un veic vienu klikšķi ar peles kreiso taustiņu, tiek iegūts riņķis.



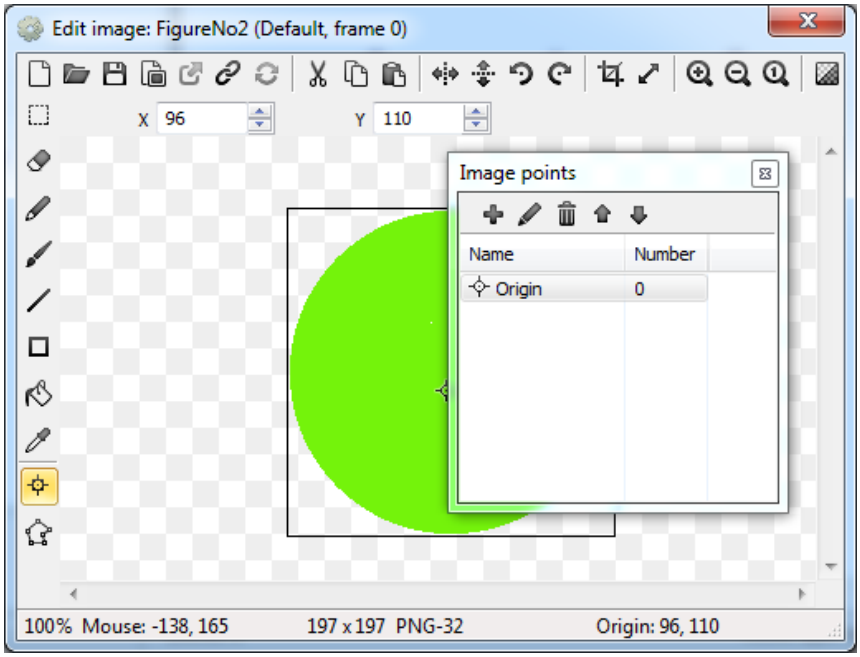

Tālāk nepieciešams apgriezt attēlu atbilstoši riņķa lielumam. Šim nolūkam izmantoi rīku **Crop transparent edges**.



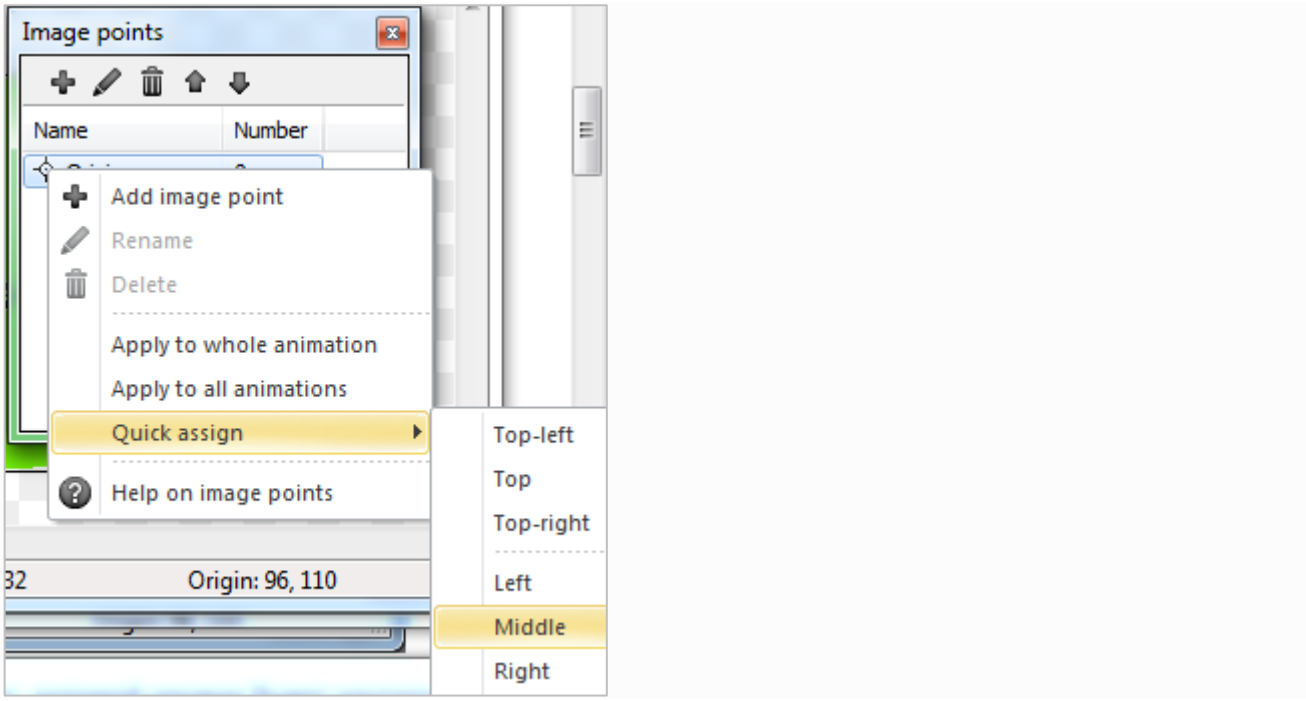



Kā redzams, pēc šīs komandas izpildes attēla izmērs ir 197x197.

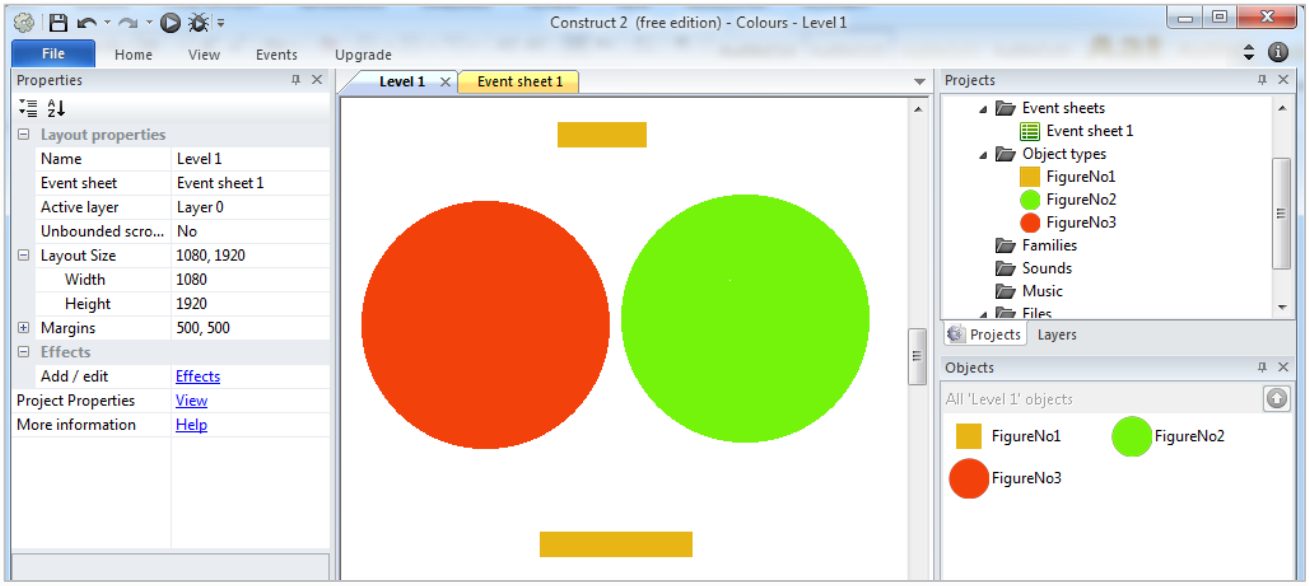
Ja tālākā spēles izstrādes gaitā plānots, ka riņķis griežas, tad jau tagad var norādīt, kur atradīsies ass (punkts), ap kuru tas griezīsies. Noklikšķina uz **Set origin and image points**.



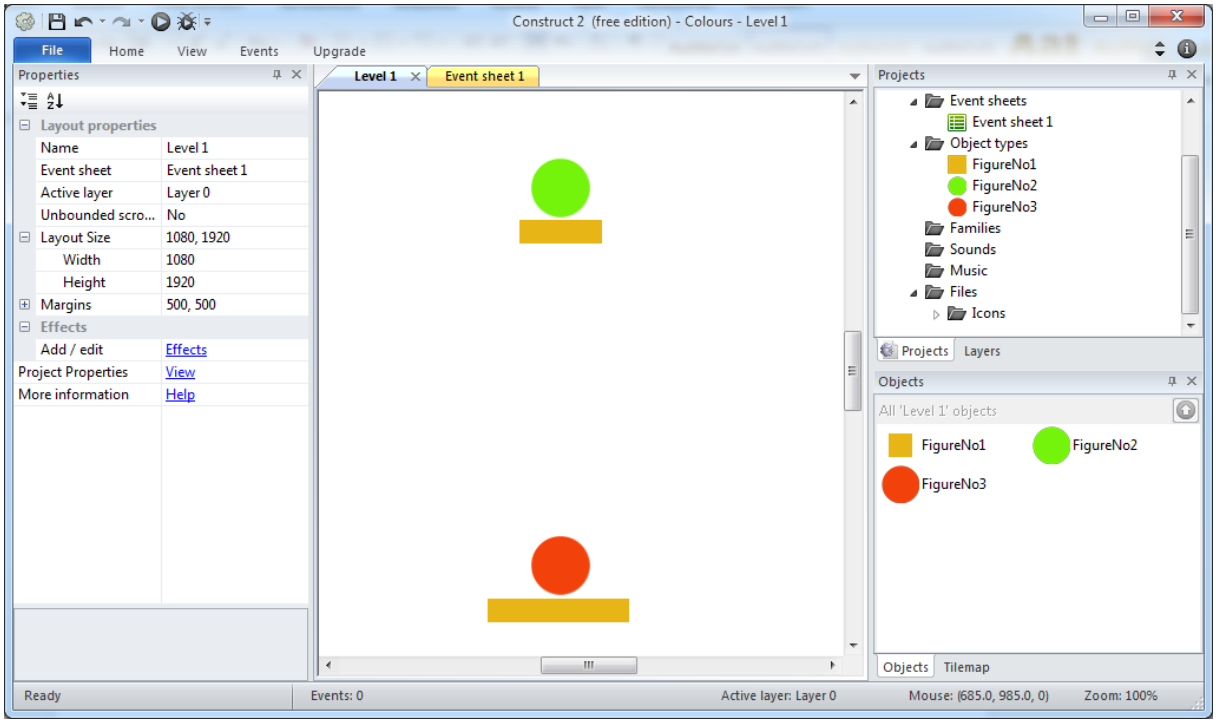
Atveras dialoga logs **Image points**, kurā ar labo taustiņu noklikšķina uz **Origin** un izvēlas **Quick assign**, pēc tam - **Middle**.

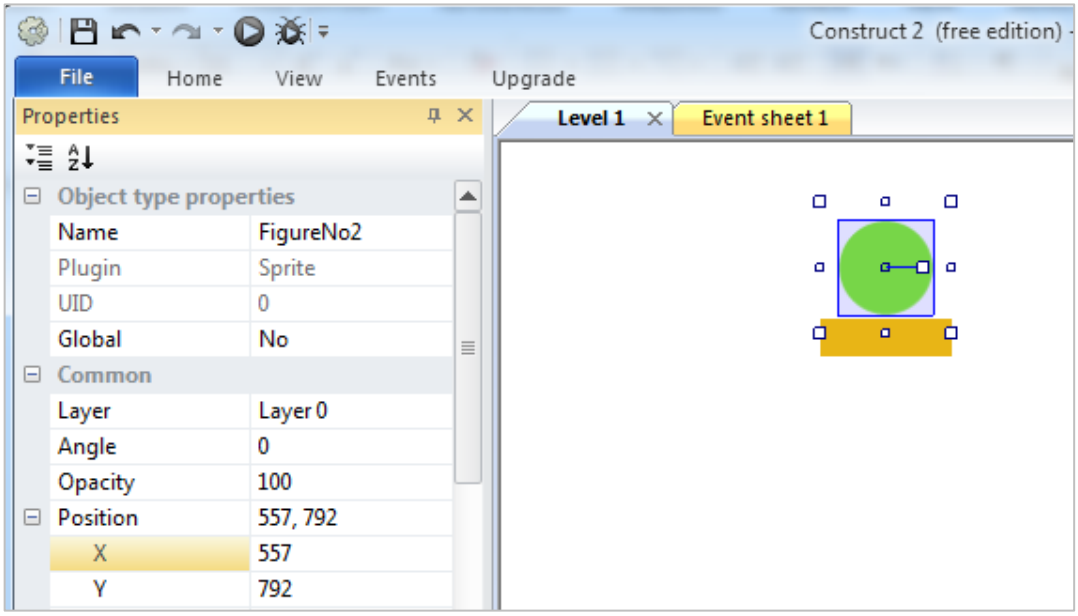



Kā jau iepriekš tika teikts, kopumā spēlē nepieciešami 2 riņķi, tāpēc izveido vēl vienu riņķi, tikai oranžā krāsā.

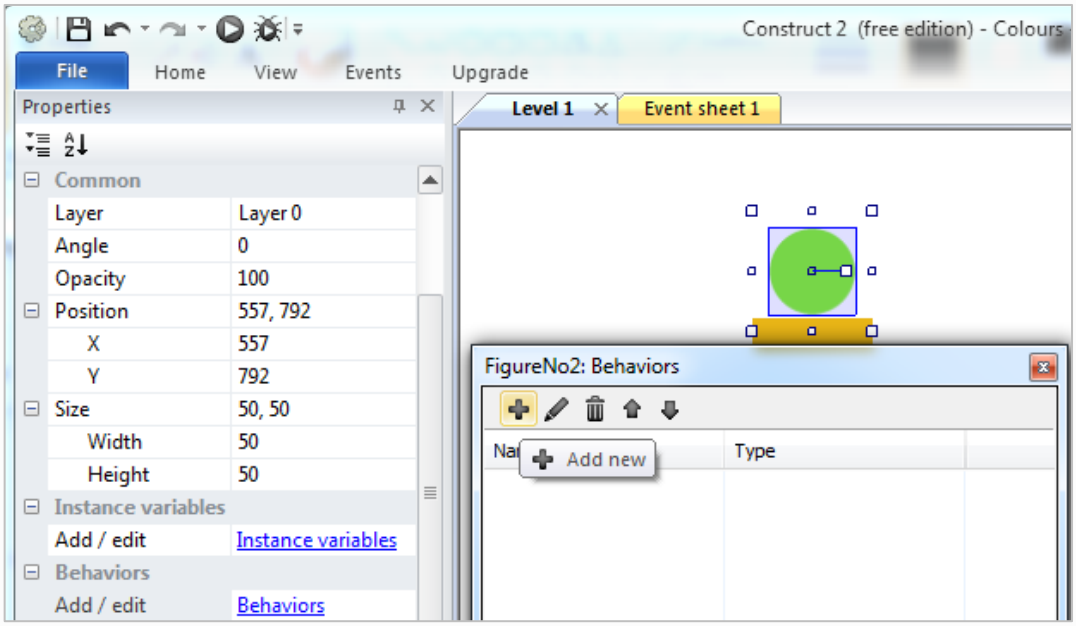


Tā kā spēlē tik lieli riņķi nebūs nepieciešami, tos samazina. Sadaļā **Properties** tiem iestata izmēru 50x50 un pēc novieto virs taisnstūriem, bet vienu riņķi tieši zem otra. Jāpievērš uzmanība, lai sakristu to koordināte x, kas konkrētajā gadījumā ir 557.

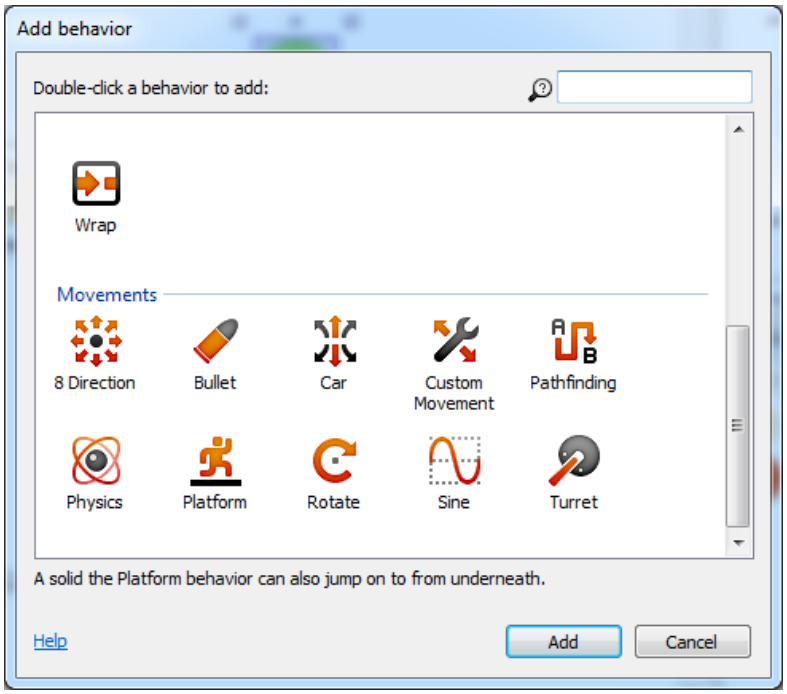


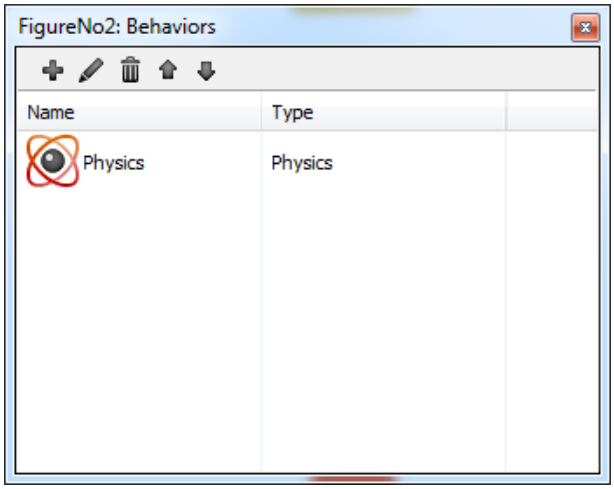



Turpinājumā jānorāda, kā riņķi kustēties. Tiek piedāvātas dažādas iespējas, bet mūsu apskatītajā variantā riņķi viens pie otra pievilksies. Izvēlas zaļo riņķi un sadaļā **Properties** noklikšķina uz **Behaviors**, atveras dialoga logs, piemēram, **FigureNo2:Behaviors**, kurā izvēlas **Add new**.



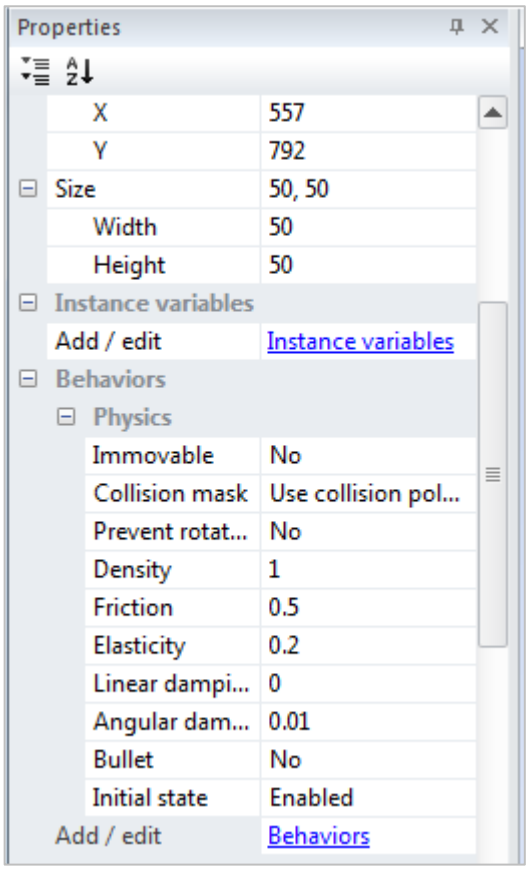

Atveras logs, kurā izvēlas sadaļu **Physics** un noklikšķina uz pogas **Add**.



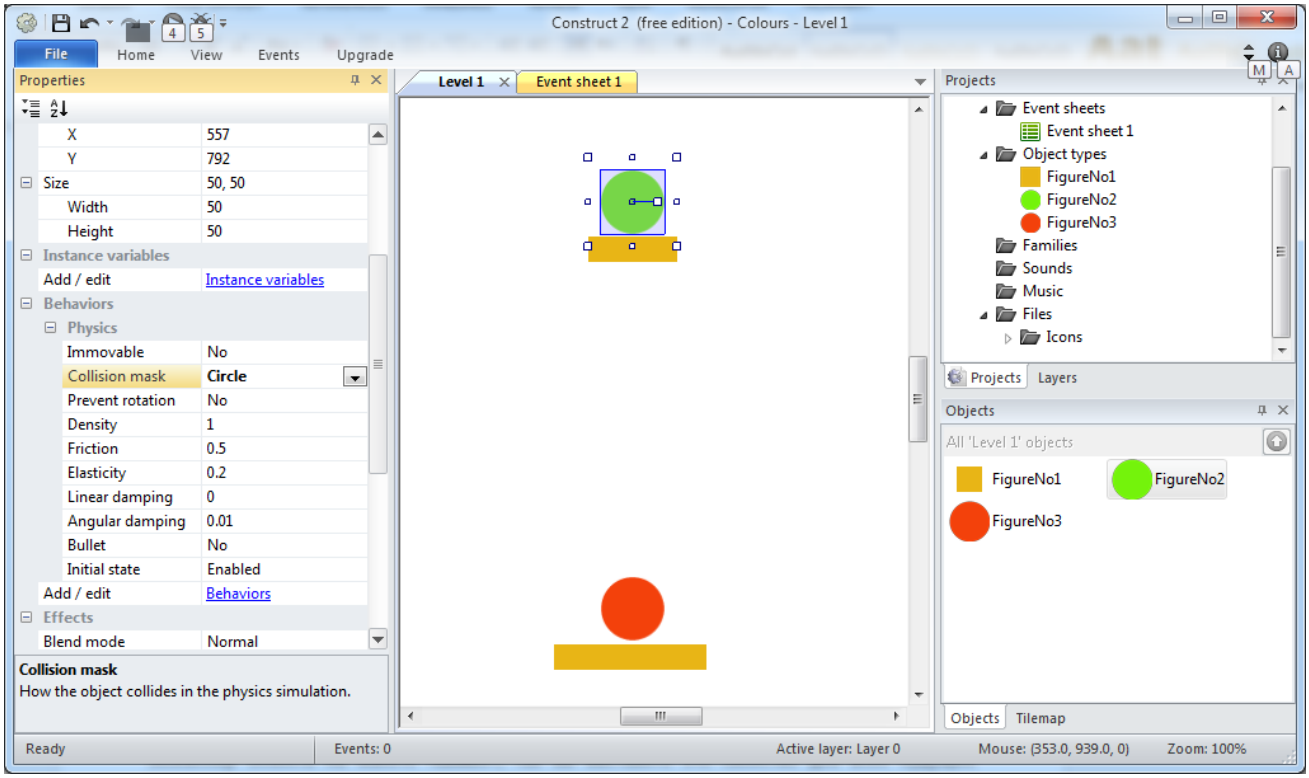



Pēc šo darbību izpildes logu var aizvērt.

Sadaļā **Properties** zem **Behaviors** tagad ir parādījusies apakšsadaļa **Physics**, kurā var norādīt dažādas īpašības.

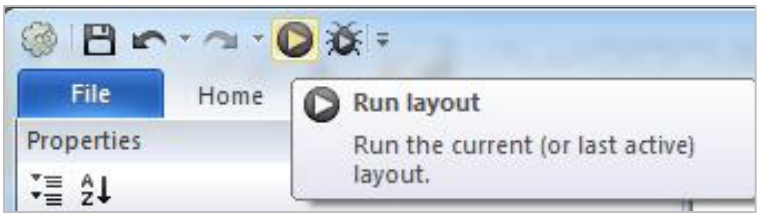


Norāda, ka mūsu figūra ir apaļa. Ierakstam **Collision mask** pretim izvēlas **Circle**.

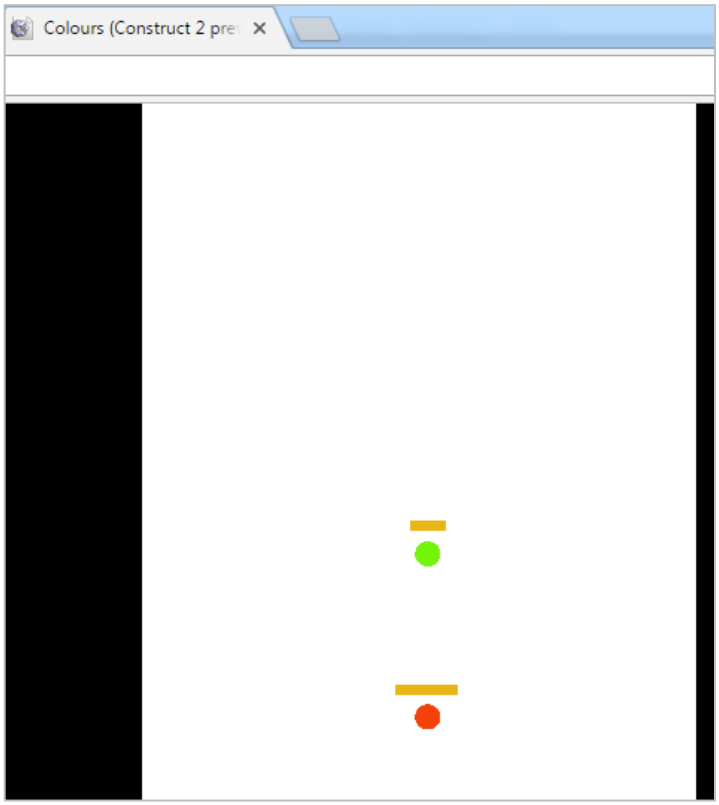



Tieši tādas pašas darbības veic arī saistībā ar oranžo riņķi.

Lai apskatītos, kā kustas riņķi, noklikšķina uz pogas **Run layout** vai arī nospiež tastatūras taustiņu **F5**.

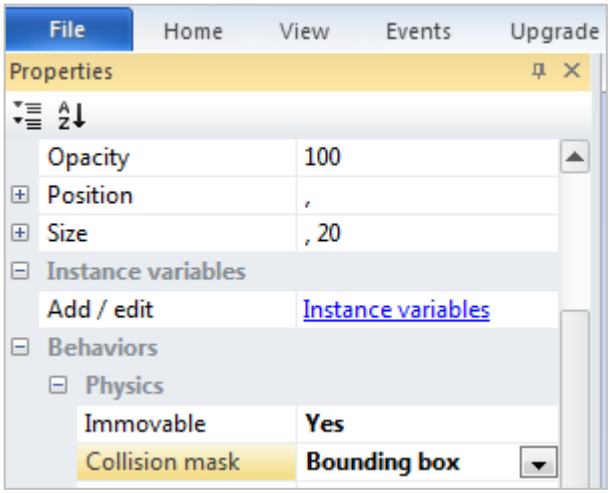



Atveras interneta pārlūkprogrammas logs, kurā redzams, ka abi riņķi krīt lejup.

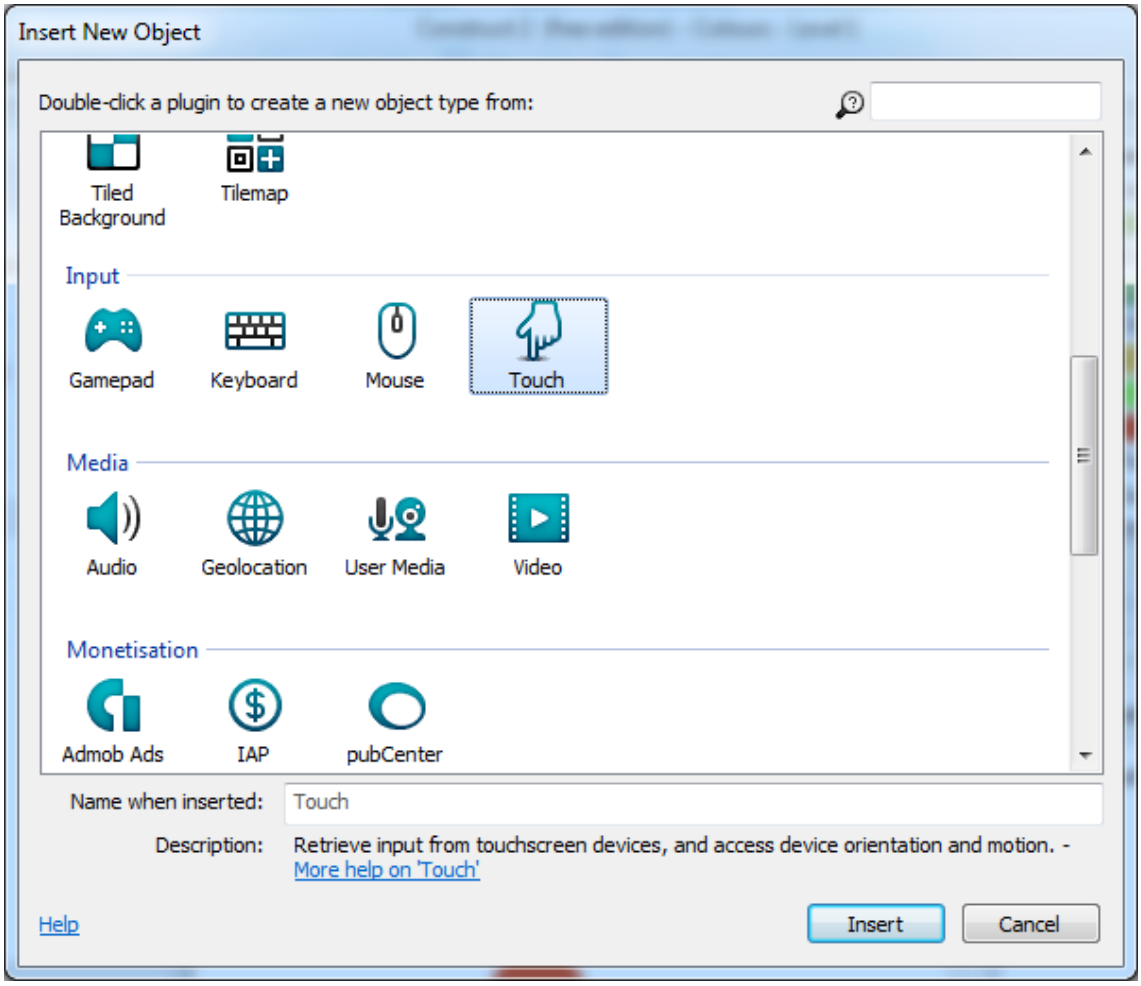



Riņķi kustas, taču pagaidām gan tie neatduras pret dzeltenajiem taisnstūriem, bet it kā pārslīd tiem pāri. Turpinājumā norāda, ka taisnstūri ir nekustīgi. Rīkojas līdzīgi, kā ar riņķiem, un sadaļā **Physics** pretim īpašībai **Immovable** norāda **Yes**. Tas nozīmē, ka figūra būs nekustīga.

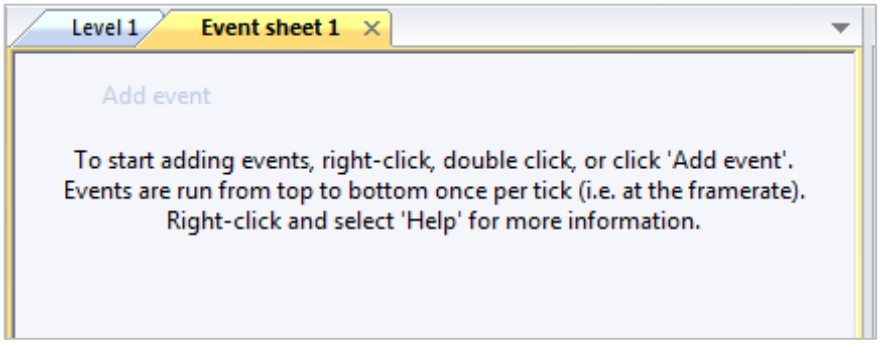
Savukārt pretim **Collision mask** izvēlas **Bounding box**. Tas nozīmē, ka objekta robežas sakritīs ar attēla robežām.



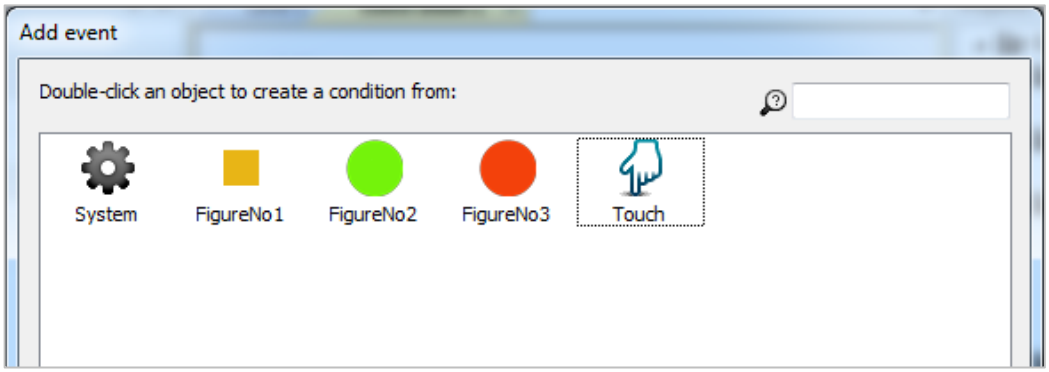
Pēc šo darbību veikšanas, apskatot spēles maketu, redzams, ka riņķi atrodas uz taisnstūriem un vairs nekrīt lejup. Turpinājumā būtu jāpanāk, lai pieskaroties dzeltenajiem taisnstūriem, tie pazustu (tos varētu noņemt). Projektam pievieno objektu **Touch**.



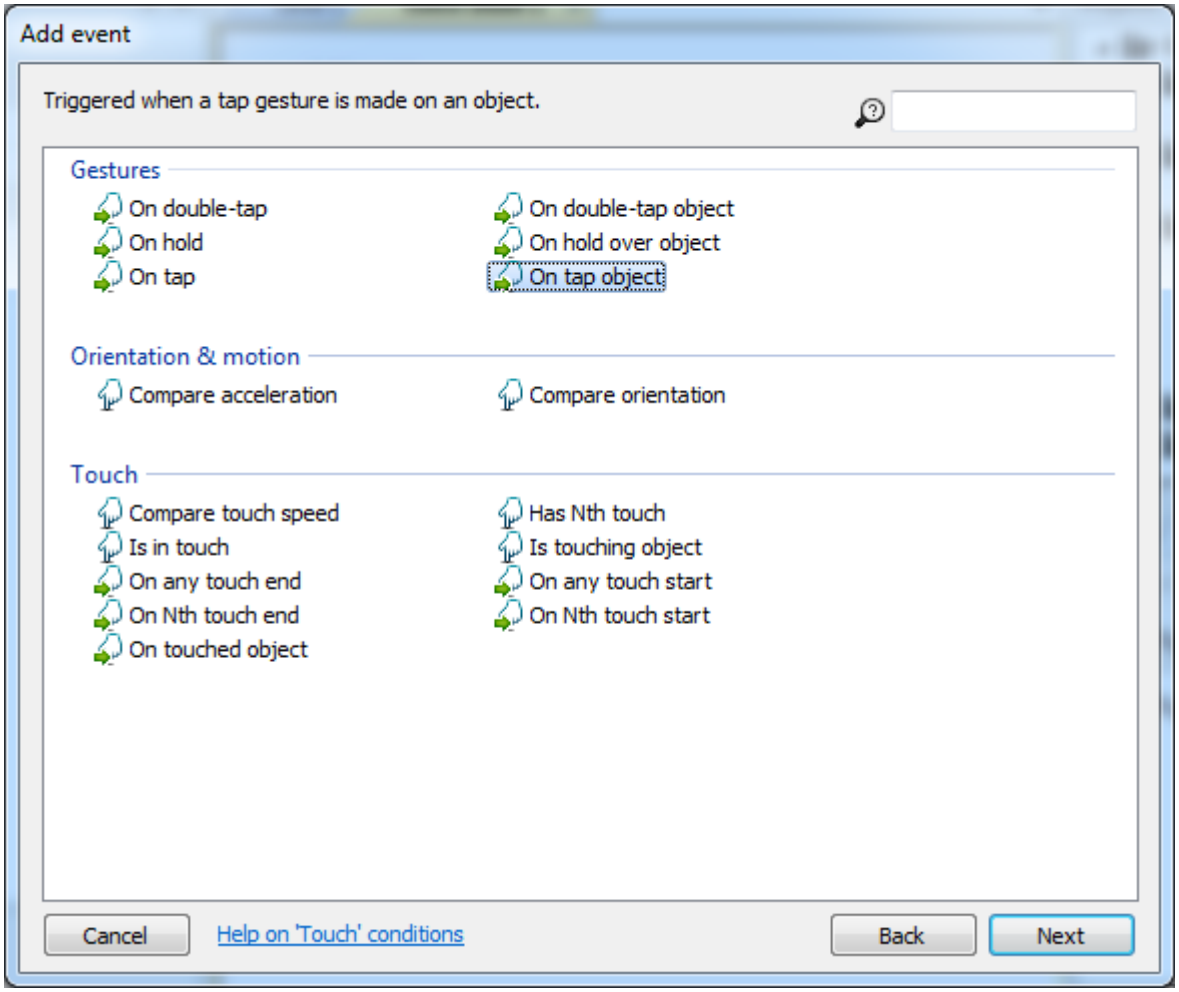

Atver cilni **Event sheet 1**. Tukšajā laukumā zemāk veic klikšķi ar peles labo taustiņu un izvēlas **Add event** vai arī izpilda dubultklikšķi.



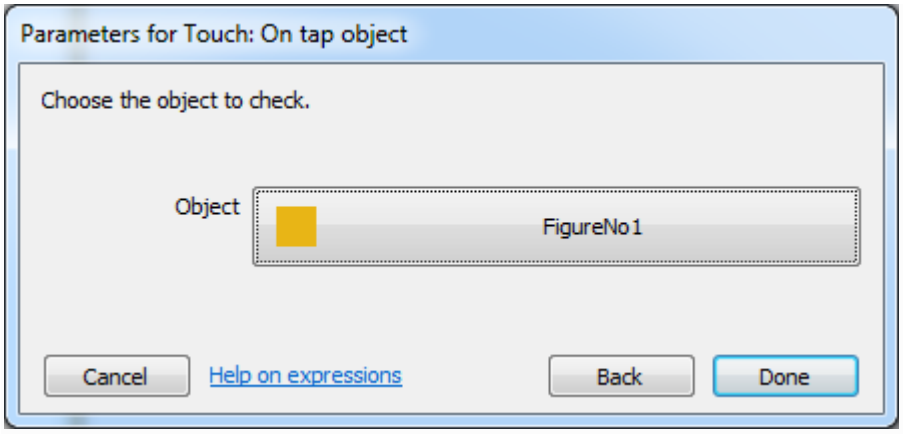
Atveras logs, kurā izvēlas objektu **Touch** un noklikšķina uz pogas **Next**.



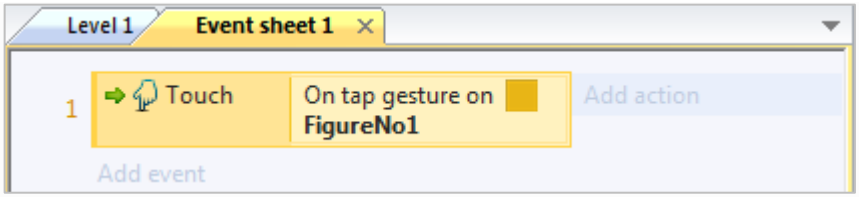

Nākošajā solī izvēlas notikumu, kuru vēlas apstrādāt. Konkrētajā gadījumā tas būs **On tap object**.



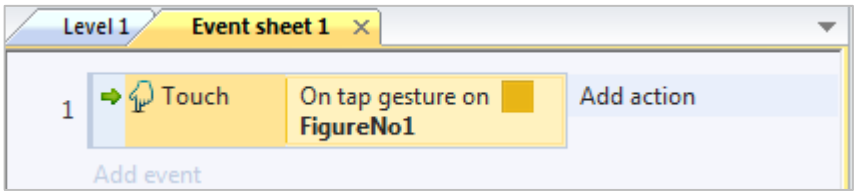
Atkal noklikšķna uz pogas **Next**. Nākošajā solī izvēlas objektu, uz kura "uzkāps" - **FigureNo1**, un noklikšķina uz pogas **Done**.



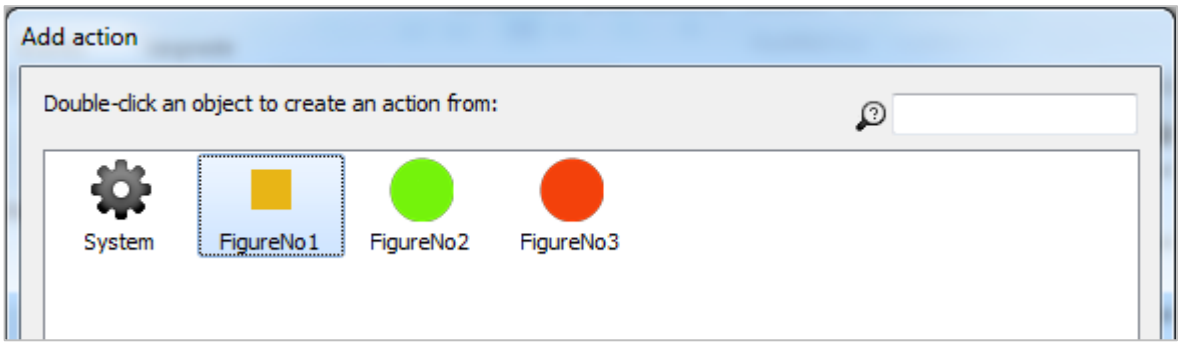

Pēc dialoga loga aizvēršanas sadaļā **Event sheet 1** redzams pirmais notikums, kas norisināsies tad, ja spēlētajs "uzkāps"uz dzeltenā taisnstūra vai arī noklikšķinās uz tā ar peli.



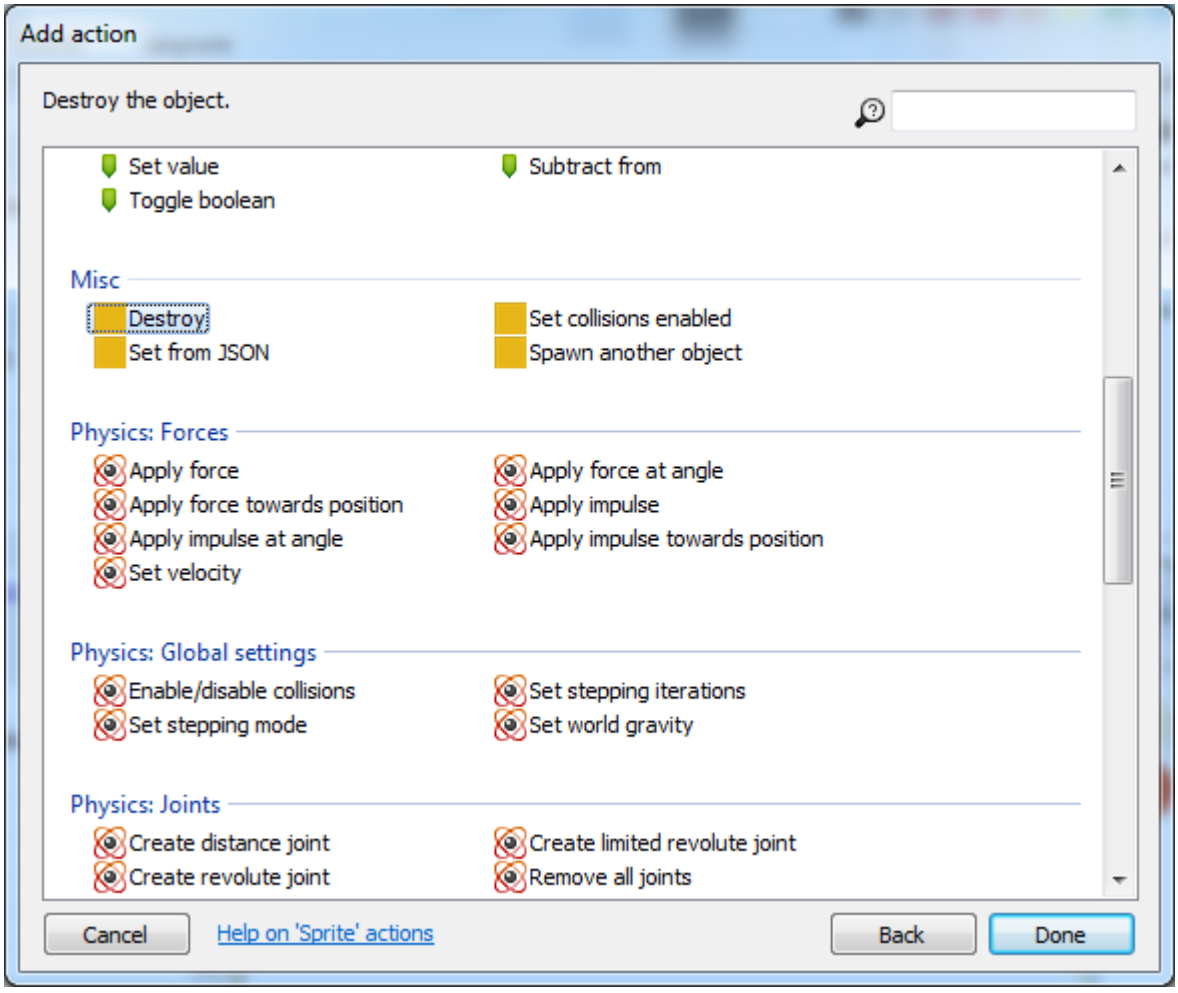

Kas notiks ar objektu, kuram uzklikšķinās vai uz kura "uzkāps"? Lai izvēlētos darbību, klikšķina uz **Add action.**



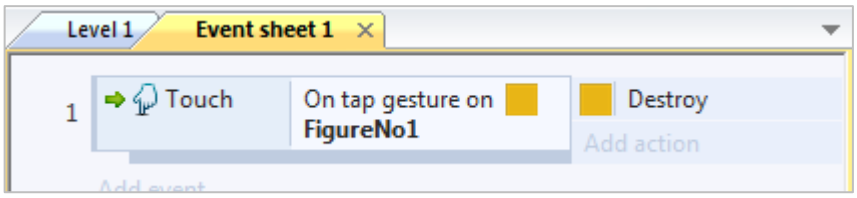

Atveras logs **Add action**, kurā izvēlas nepieciešamo objektu un noklikšķina uz pogas **Next**.



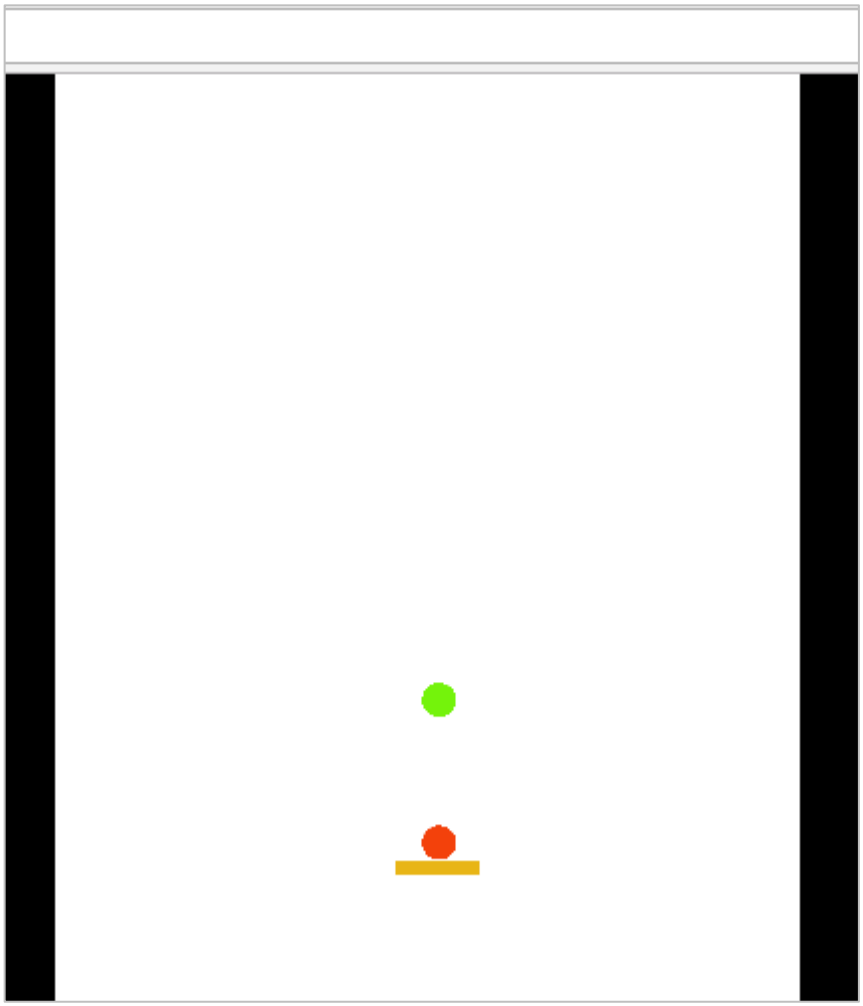

Nākošajā solī jāizvēlas kāda konkrēta darbība. Tā būs objekta dzēšana, tāpēc izvēlas **Destroy** un noklikšķina uz pogas **Done**.




Pēc loga aizvēršanas notikumu sadaļā redzama jaunā darbība.

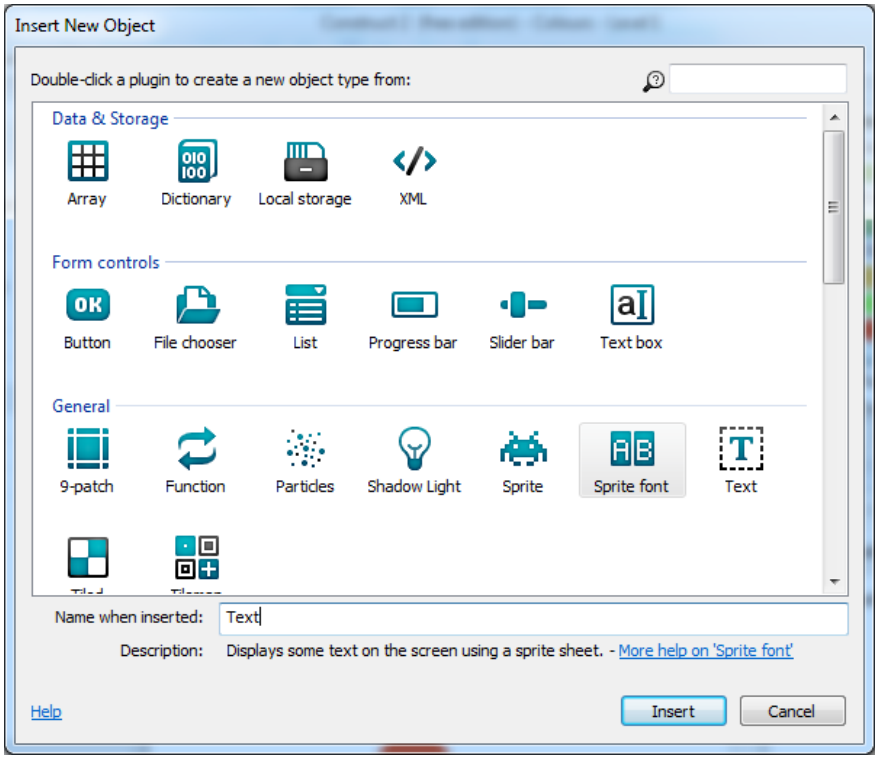



Noklikšķinot uz **Run layout,** var apskatīties, kā tagad kustēsies figūras. Sākumā redzami divi dzelteni taisnstūri, uz kuriem atrodas riņķi. Noklikšķinot uz augšējā taisnstūra, tas pazūd un riņķis sāk krist lejup.

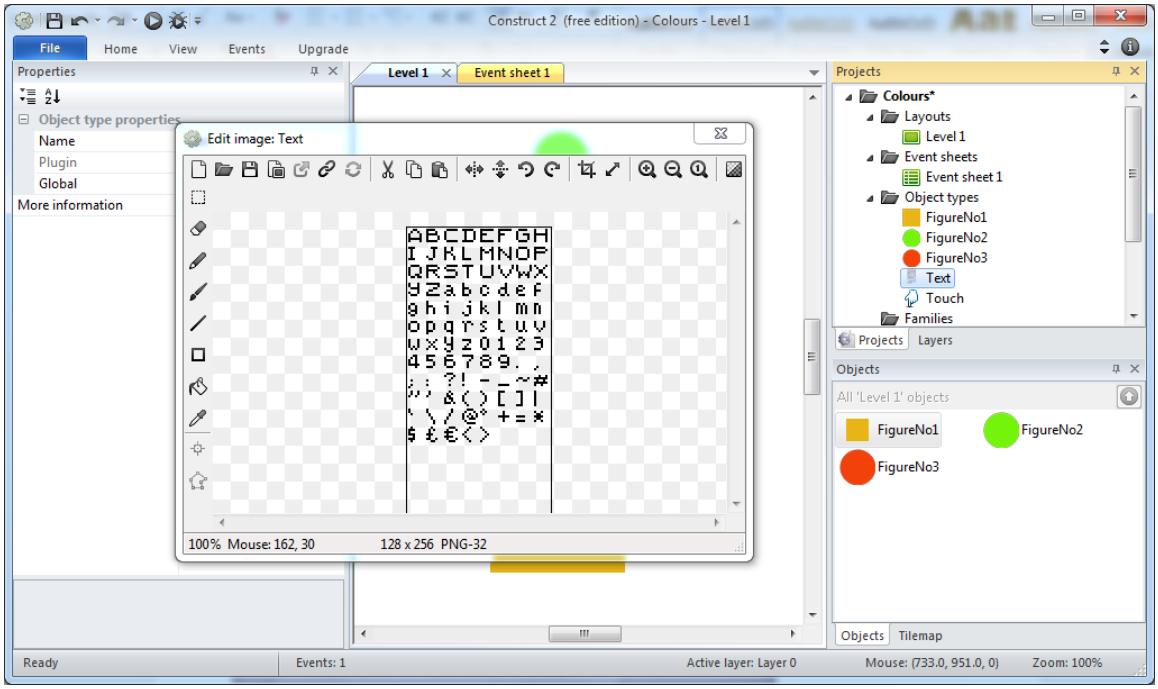


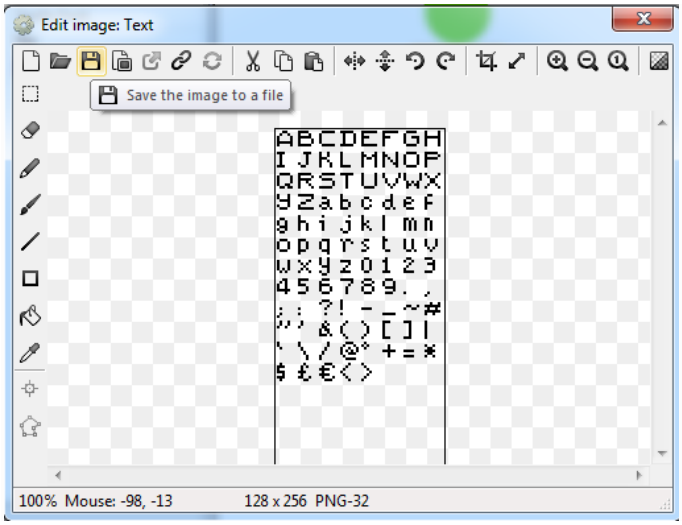
Programmā **Construct** pēc vajadzības ir iespējams pievienot arī dažādus tekstus, kas parādās ekrānā, kad veikts kāds uzdevums.

Atver dialoga logu **Insert new object**, izvēlas **Sprite font**, piešķir jaunajam objektam nosaukumu **Text** un noklikšķina uz **Insert**.




Atveras jauns logs **Edit image**, kurā tiek piedāvāti dažādi burtu fonti. Ja logs neatveras automātiski, tad to var aktivizēt sadaļā **Projects,** veicot dubultklikšķi uz objekta nosaukuma **Text.**

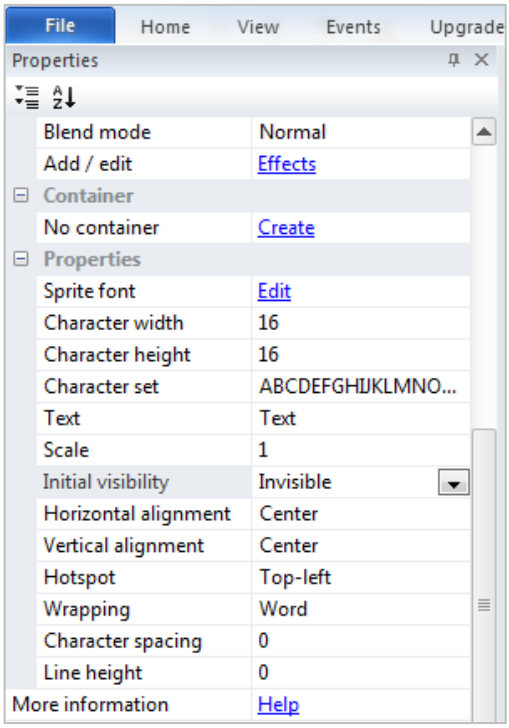


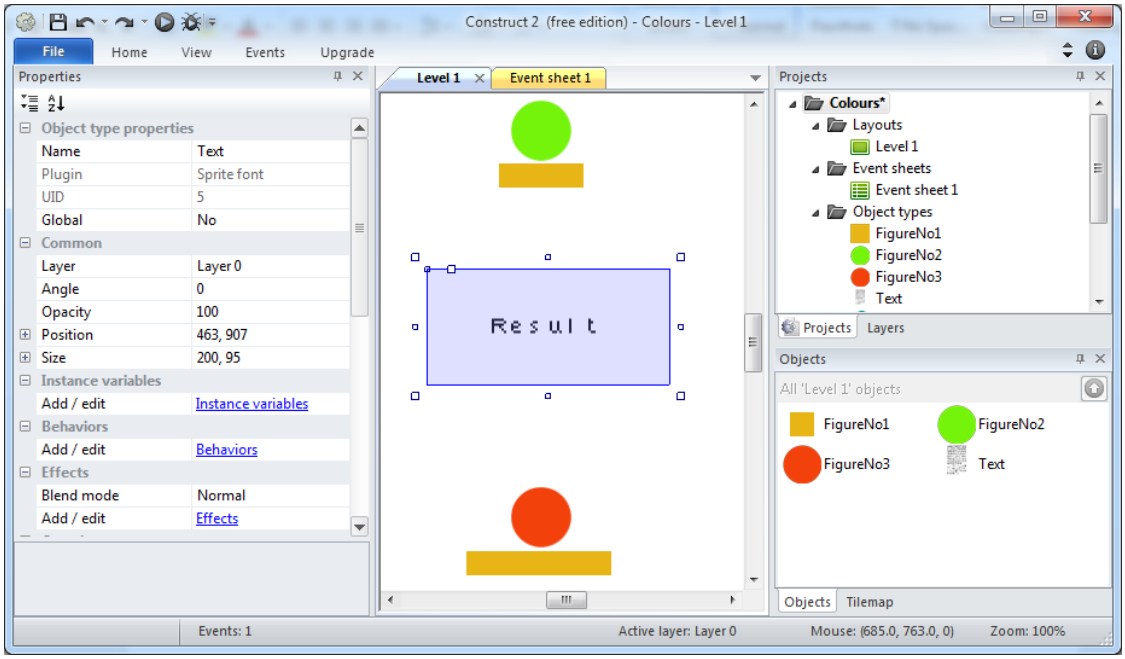



Objektā **Sprite** font teksta attēlošanai tiek piedāvāti šajā attēlā redzamās rakstu zīmes. Lai attēlu saglabātu datorā, noklikšķina uz **Save the image to a file**. Pēc saglabāšanas logu **Edit image** var aizvērt.

Objekta **Sprite font** īpašību sadaļā var norādīt, piemēram, rakstzīmju platumu, augstumu, teksta līdzinājumu u.tml.

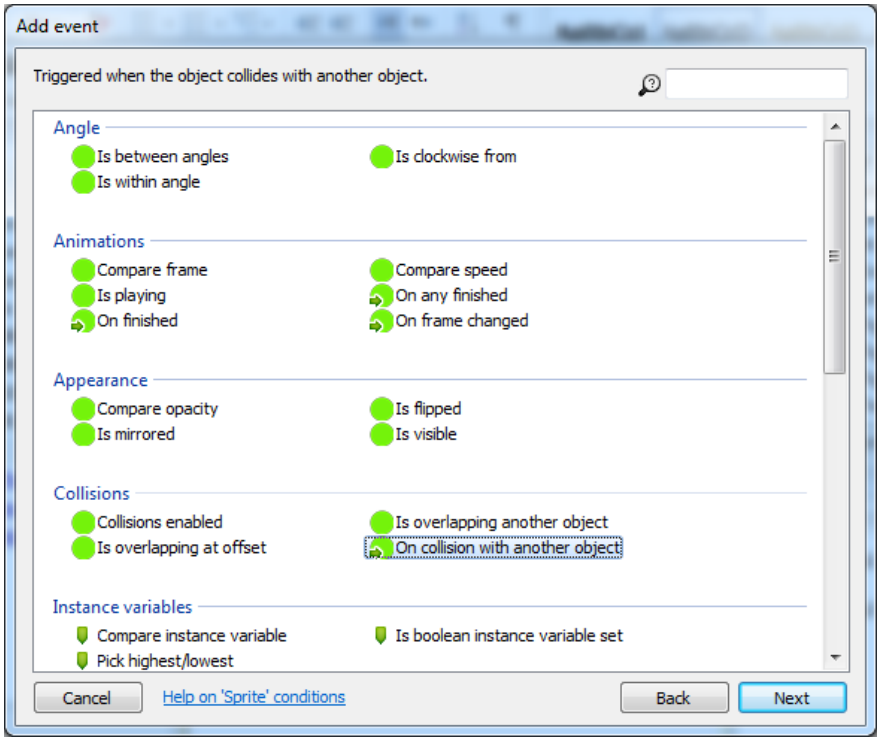
Ja vēlas, lai teksts sākumā nebūtu redzams, pretim **Initial visibility** izvēlas **Invisible**.





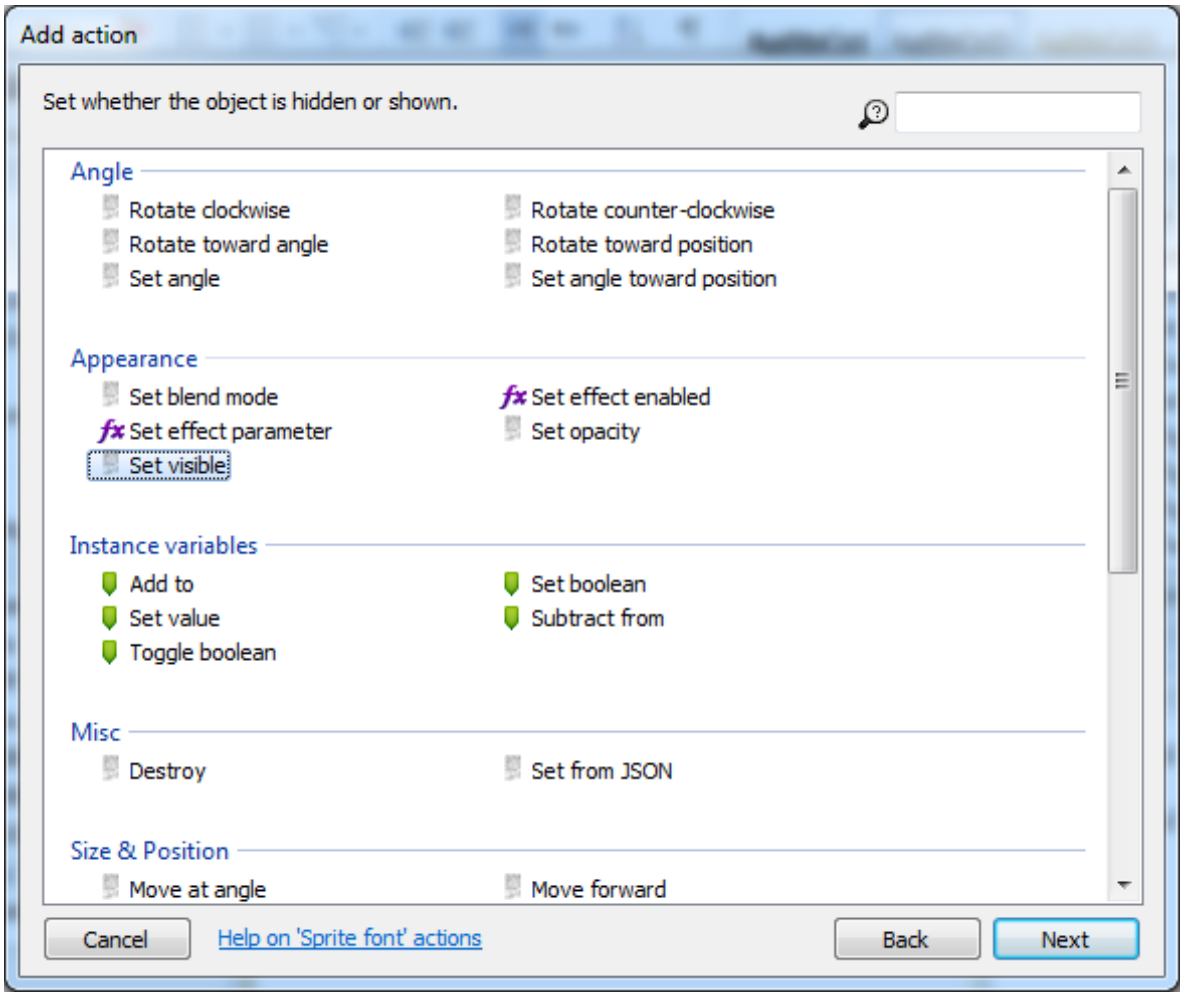

Turpinājumā jānorāda, kas notiks ar zaļo riņķi, kad tas uzkritīs uz oranžā riņķa. Šim nolūkam atkal pievieno jaunu notikumu, atlasa zaļo riņķi un noklikšķina uz pogas **Next**.

Atveras logs **Add event**, kurā izvēlas notikumu **On collision with another object** un noklikšķina uz **Next**.



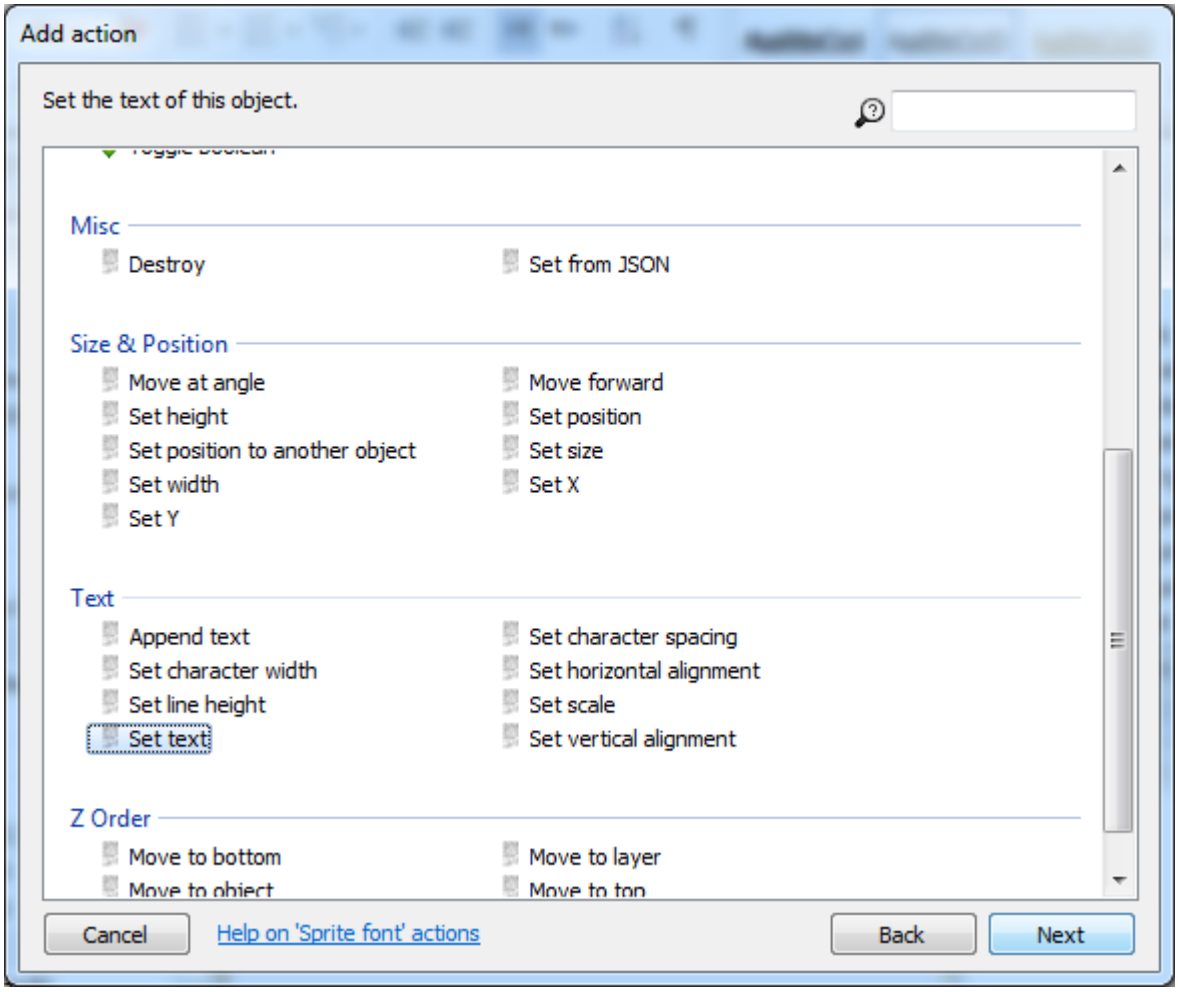
Pēc tam logā **Object** izvēlas oranžo riņķi, ar kuru notiks sadursme, un noklikšķina uz **Done**.

Kad notikums pievienots, noklikšķina uz **Add action**, izvēlas objektu **Text** un noklikšķina uz pogas **Next**. Atveras logs **Add action**, kurā izvēlas **Set visible** un noklikšķina uz **Next**.

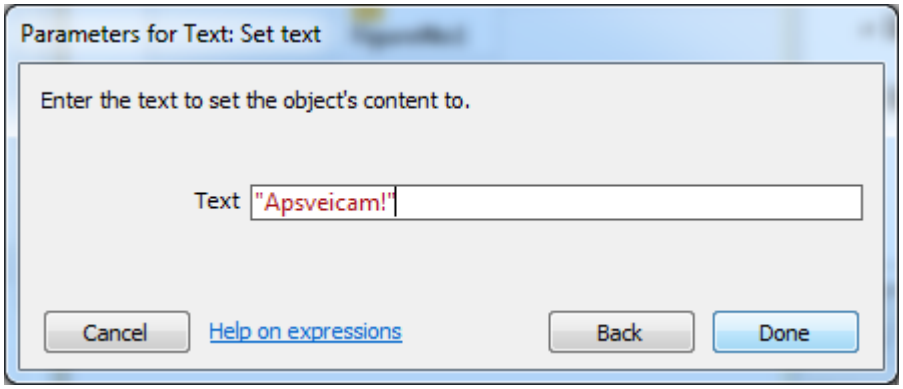



Nākošajā solī izvēlas vērtību **Visible** un noklikšķina uz **Done**.

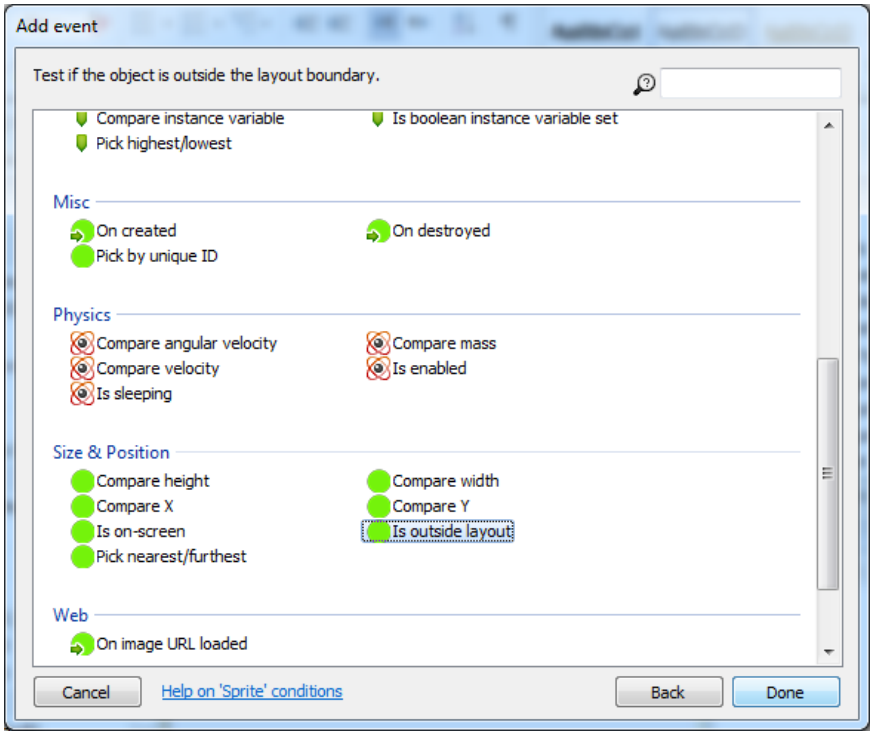
Noslēgumā var pievienot arī apsveikuma tekstu. Noklikšķina uz **Add action** un izvēlas objektu **Text**, pēc tam darbību **Set text**. Kad tas paveikts, noklikšķina uz pogas **Next**.



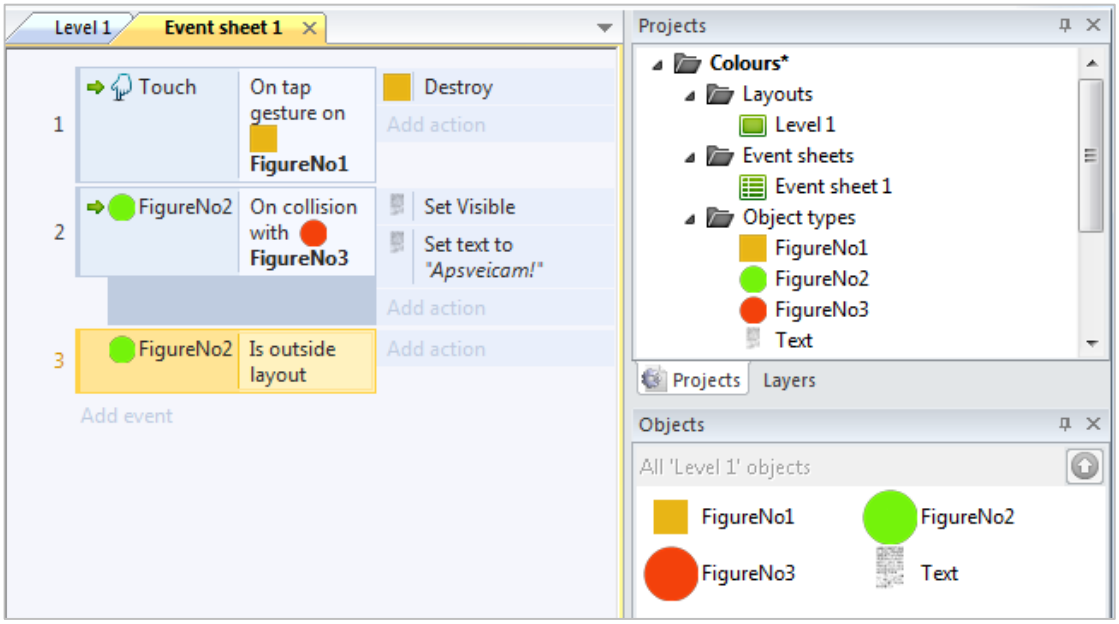

Ievada apsveikuma tekstu, piemēram, **“Apsveicam!”** un noklikšķina uz **Done**.



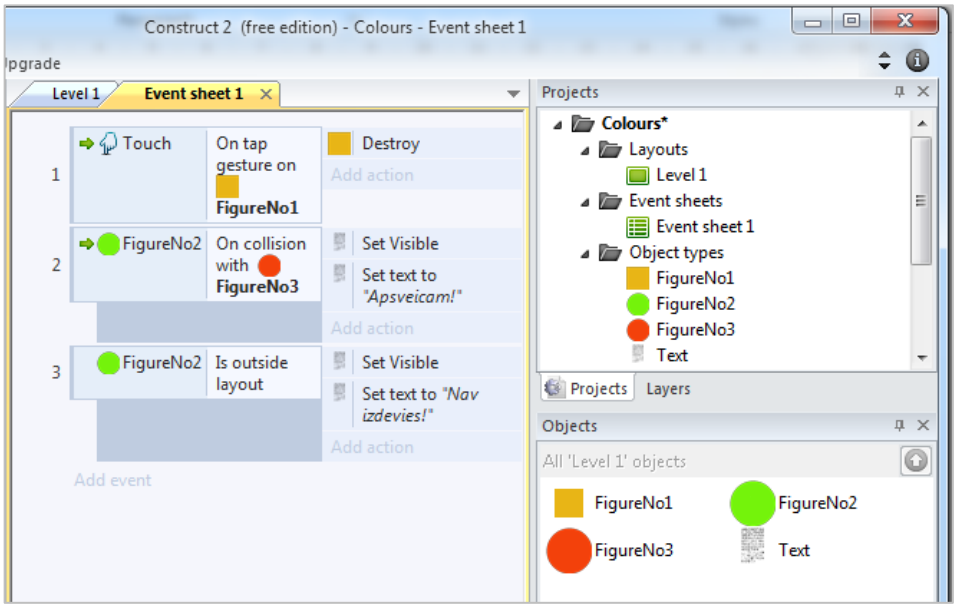
Turpinājumā vajadzētu noformēt paziņojumu, ka 1. līmeņa uzdevums nav veikts, t.i., ja riņķi izslīd ārpus maketa. Zaļajam riņķim pievieno jaunu aktivitāti, noklikšķinot uz **Add event**, pēc tam atlasot zaļo riņķi, noklikšķinot uz **Next** un izvēloties **Is outside layout**.




Pēc šo darbību izpildes notikumu sadaļai vajadzētu izskatīties šādi:

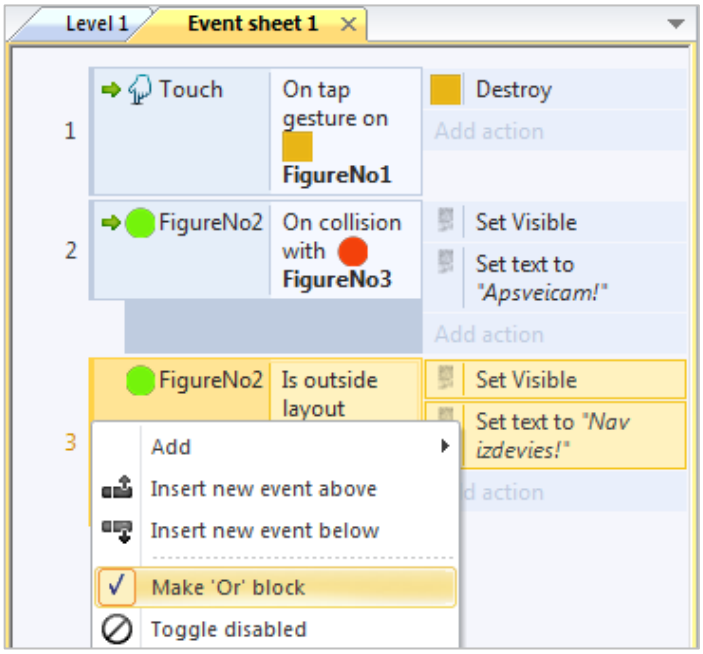


Pēc tam pievieno tekstu.

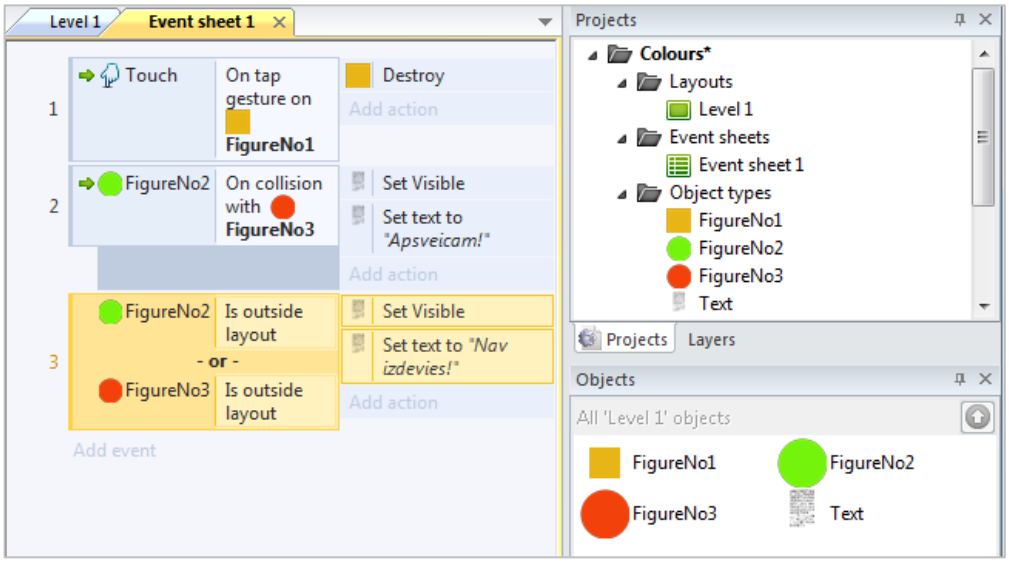



Oranžā riņķa darbības būs tādas pašas, tāpēc pie zaļā riņķa uz notikuma **Is outside layout** noklikšķina ar peles labo taustiņu uz izvēlas **Add another conditions**, pēc tam pievieno tādas pašas darbības arī oranžajam riņķim.

Nepieciešams panākt, lai darbības izpildītos saistībā ar jebkuru notikumu, tāpēc veic klikšķi ar peles labo taustiņu vairāk pa kreisi, apmēram vietā, kur ir notikumu numerācija un izvēlas **Make 'Or' block**.



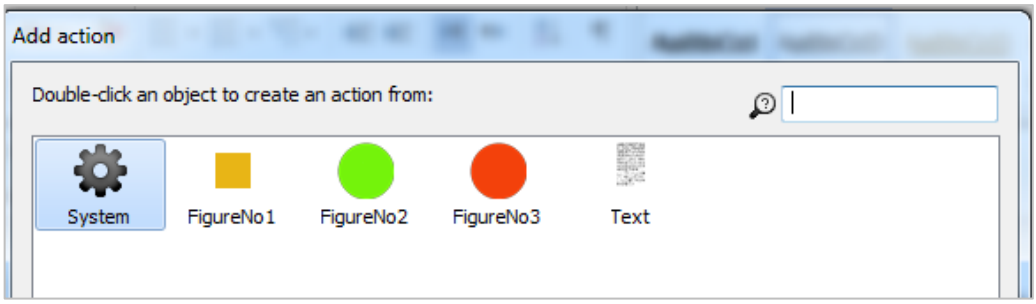
Tiks izveidots bloks.



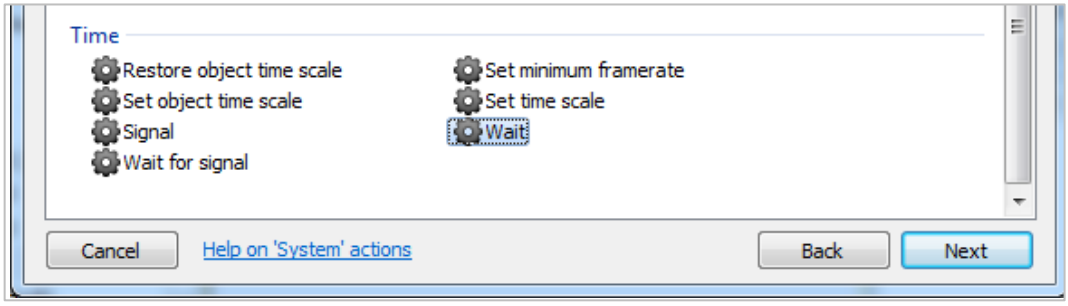

Tagad var apskatīties, kā spēle darbojas. Piemēram, uzklikšķinot uz augšējā dzeltenā taisnstūra, parādīsies uzraksts "Apsveicam!", ja uz apakšējā – "Nav izdevies!"

Spēlei var pievienot vēl citas papildus funkcijas, piemēram, kad noteikts līmenis pabeigts, pēc dažām sekundēm tiek automātiski piedāvāts spēlēt atkal no jauna šo vai, ja ir, citu līmeni.

Klikšķina uz **Add action**. Atveras logs, kurā klikšķina uz **System** un pēc tam uz pogas **Next**.

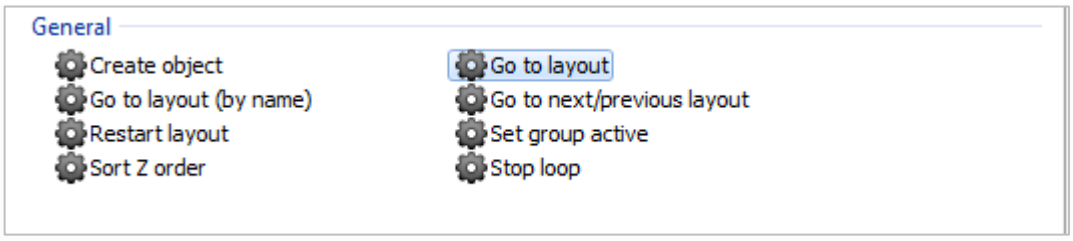



Nākošajā solī sadaļā **Time** izvēlas **Wait** un atkal klikšķina uz **Next**.

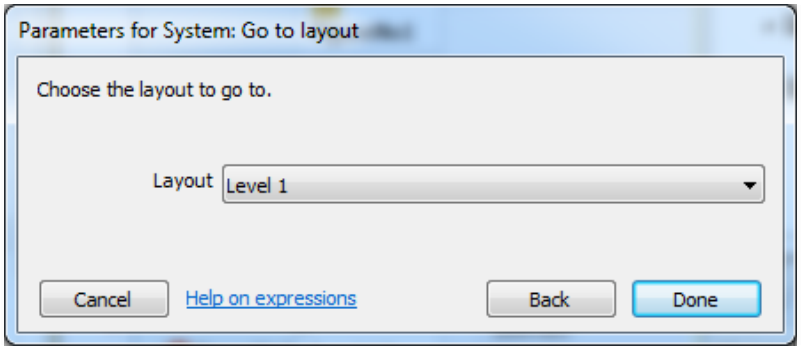


Iestata laiku, piemēram, 4 sekundes un noklikšķina uz **Done**.

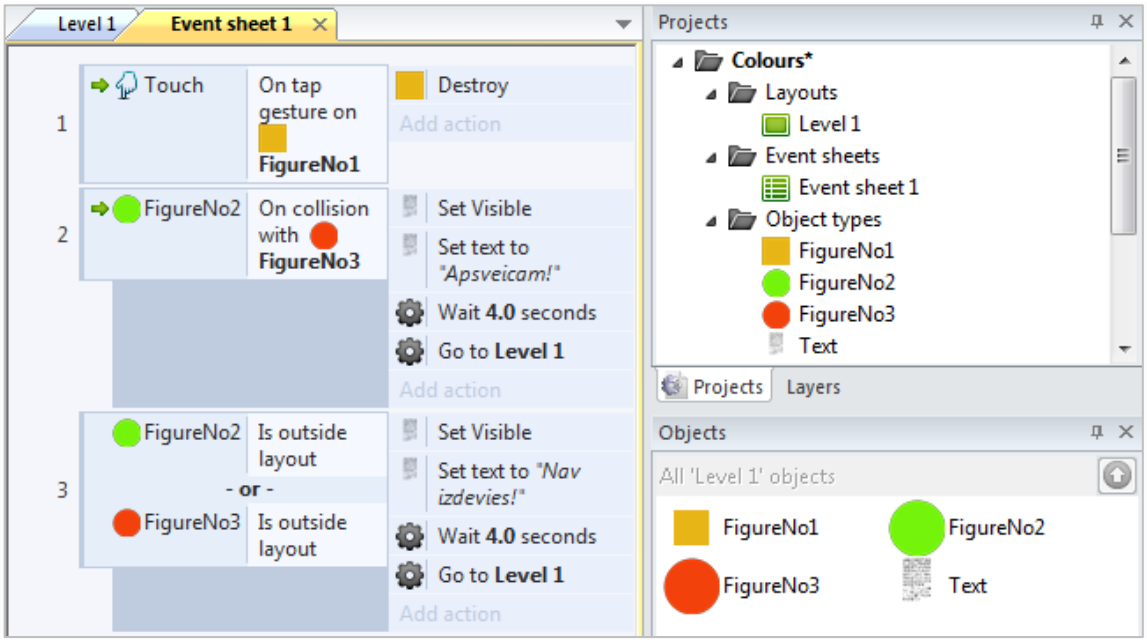
Pēc tam klikšķina vēlreiz uz **Add action**, **System** un sadaļā **General** izvēlas **Go to Layout**, pēc tam noklikšķina uz **Next**.




Nākamajā solī norāda līmeni, uz kuru jāpāriet. Konkrētajā gadījumā ir tikai viens līmenis, tāpēc izvēlas **Level 1** un noklikšķina uz **Done**.




Rezultātam vajadzētu izskatīties šādi. Tagad var vēlreiz pārbaudīt, vai viss darbojas.



# Autortiesības

**Autortiesības** ir autora tiesiskais stāvoklis, kas nosaka sevišķas tiesības publicēt, reproducēt un izplatīt oriģināldarbu (literāru, muzikālu, mākslas utml. darbu, programmas pirmkodu, audiovizuālos materiālus, attēlus, tekstus utt.) jebkurā vidē.

**Autortiesību likums**[6] nosaka, ka:

- Autortiesības pieder autoram, tiklīdz darbs ir radīts, neatkarīgi no tā, vai darbs ir pabeigts.
- Autortiesības attiecināmas uz literāriem, zinātnes, mākslas un citiem šā likuma darbiem, arī nepabeigtiem darbiem, neatkarīgi no darba uzdevuma un vērtības, izpausmes formas vai veida.
- Autortiesību piederības apliecināšanai nav nepieciešama reģistrācija, darba speciāla noformēšana vai kādu citu formalitāšu ievērošana.
- Autors vai viņa tiesību pārņēmējs savas tiesības uz darbu var apliecināt ar autortiesību aizsardzības zīmi, kuru atveido ar tādu paņēmienu un tādā vietā, lai tā būtu nepārprotami redzama. Šī zīme ietver trīs elementus:
  - burtu "C" aplītī;
  - autortiesību subjekta vārdu (nosaukumu);
  - darba pirmpublicējuma gadu.

**Bieži vien, veidojot kādu darbu, rodas jautājums, vai savos materiālos drīkst izmantot interneta resursus.**

To drīkst darīt, ja autors ir devis atļauju izmantot viņa darbu.

Kā minēts "Eversheds Bitāns" skaidrojumā[7], "Šādu atļauju var saņemt, noslēdzot licences līgumu vai saņemot licenci. Jāpiebilst, ka licences līgums var tikt slēgts gan rakstveidā, gan mutiski".

Cita autora darbu drīkst izmantot arī, ja šādas tiesības jums piešķir īpaša licence. Piemēram, ***Creative Commons*** licence sniedz iespēju izmantot resursus atbilstoši dažiem nosacījumiem.

**Kas jāievēro mācību līdzekļa autoram, ievietojot savā mācību līdzeklī citu autoru aizsargājamus darbus vai to fragmentus?**

Atsaucoties uz "Eversheds Bitāns" skaidrojuma par autortiesību un blakustiesību objektu izmantošanu mācību procesā, "Pirmkārt, svarīgākais noteikums, kas jāievēro, izmantojot citu autoru aizsargātus darbus vai to fragmentus paša radītā mācību līdzeklī, ir atsaukšanās uz šī darba autoru un šī darba nosaukumu, kā arī publicēšanas datiem vai citu avotu, kur attiecīgais darbs pieejams vai apskatāms u.tml.

---

[6] https://likumi.lv/doc.php?id=5138
[7] http://www.visc.gov.lv/vispizglitiba/saturs/dokumenti/20131202_skaidrojums_par_autortiesibam.pdf

Otrkārt, rūpīgi jāizvērtē nolūks, kādā izmantoti citu autoru darbi vai to fragmenti.

Likums paredz izņēmumu no autoru reproducēšanas tiesībām, kad šādi darbi vai to fragmenti tiek reproducēti nolūkā radīt jaunu mācību līdzekli – filmu, raidījumu, plakātu, uzdevumu, spēli u.tml. It sevišķi, ja šie citu autoru darbi vai to fragmenti izmantoti nolūkā ilustrēt apgūstamo vielu. Tādējādi var atsaukties uz izņēmumu, kas paredz tiesības izmantot darbu izglītības mērķiem (neprasot autora piekrišanu un nemaksājot atlīdzību). Tomēr jāievēro, ka šādā gadījumā radītie mācību līdzekļi izmantojami tikai mācību iestādē (tātad nav atļauts tos padarīt pieejamus internetā), turklāt šādai izmantošanai jābūt nepastarpinātā mācību procesā un bez komerciāla nolūka. Ja mācību līdzekļa autors vēlas radīto mācību līdzekli publicēt, izmantojot trešo personu pakalpojumus (piemēram, izdevniecību) un/vai radīto mācību līdzekli vai tā kopijas izplatīt par atlīdzību vai padarīt tās pieejamas sabiedrībai internetā, tādā gadījumā mācību līdzekļa autors nevar atsaukties uz izņēmumu no autora tiesībām izglītības mērķiem, un viņam ir jāsaņem attiecīgo autoru piekrišana. Jāatceras, ka jebkurā gadījumā – arī radot jaunu mācību līdzekli, kurā citu autoru darbi izmantoti, lai ilustrētu mācāmo vielu - šādas izmantošanas apjoms nedrīkst būt pretrunā ar darba normālu izmantošanu (nedrīkst ekonomiski konkurēt ar to) un nedrīkst nepamatoti ierobežot autora likumīgās intereses. Gadījumā, ja šie citu autoru darbi vai to fragmenti izmantoti, citējot tos, tad var atsaukties uz izņēmumu no autora tiesībām, kad darbi tiek izmantoti informatīvam mērķim."[4]

### Creative Commons licence

Creative Commons ir nevalstiska bezpeļņas organizācija, kas strādā, lai palīdzētu radošumam mazināt juridiskos šķēršļus, izmantojot jaunus likumus un jaunas tehnoloģijas. Creative Commons licences pašlaik ir pieejamas vairāk kā piecdesmit valstīs visā pasaulē. Creative Commons International vietnē iespējams apskatīt šo licenču pašreizējo statusu katrā valstī.

# Izmantoto informācijas resursu saraksts


*20 Best Programming Languages In 2017* [tiešsaiste] [skatīts 2017. g. 20. martā]. https://www.techworm.net/2017/03/top-20-popular-programming-languages-2017.html

*Computer game development* [tiešsaiste] [skatīts 2017. g. 25. martā]. http://www.obscure.co.uk/articles-2/computer-game-development-an-overvie

*FREE Game Templates in Construct 2* [tiešsaiste] [skatīts 2017. g. 12. martā]. *https://blogs.msdn.microsoft.com/gamewords777/2014/09/25/free-game-templates-in-construct-2/*

*Scirra* [tiešsaiste] [skatīts 2017. g. 15. martā]. https://www.scirra.com/

*The Process of Game Creation & the Game Design Document* [tiešsaiste] [skatīts 2017. g. 10. martā].

https://digitalworlds.wordpress.com/2008/04/10/the-process-of-game-creation-the-game-design-document/

*Autortiesību likums* [tiešsaiste] [skatīts 2017. g. 27. martā]. https://likumi.lv/doc.php?id=5138

*Pirmās datorspēles* [tiešsaiste] [skatīts 2017. g. 9. martā]. http://www.sakaru-pasaule.lv/main.php3?sub=view&RID=1497

*Programmēšanas valodu tops* [tiešsaiste] [skatīts 2017. g. 9. martā]. https://pods.lv/2013/04/07/programmesanas_valodu_tops_2013_gada_marta/

*Что такое Construct 2?* [tiešsaiste] [skatīts 2017. g. 25. martā]. http://c2community.ru/info/construct2.html

*Использование звуков и музыки в Scirra Construct #2* [tiešsaiste] [skatīts 2017. g. 16. martā]. https://gcup.ru/publ/engines/ispolzovanie_zvukov_i_muzyki_v_scirra_construct_2/2-1-0-202

*Руководство к программе Construct 2 начального уровня* [tiešsaiste] [skatīts 2017. g. 12. martā]. *https://www.scirra.com/tutorials/37/construct-2/ru/page-7#h2a23*

© 2017