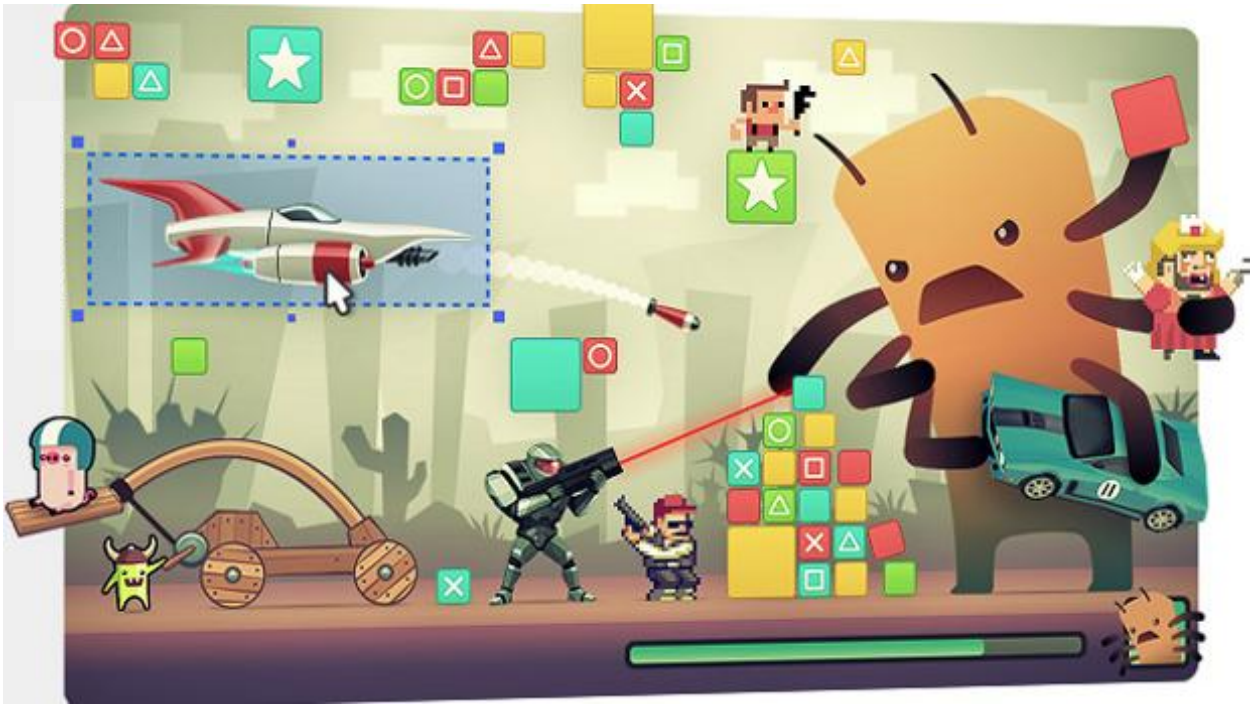
Modul Praktikum

Game Development



Program Studi Teknik Informatika

STMIK STIKOM Indonesia

DAFTAR ISI

MODUL I

MENGENAL GAME ENGINE CONSTRUCT 2

(Pertemuan 4)

Tujuan:

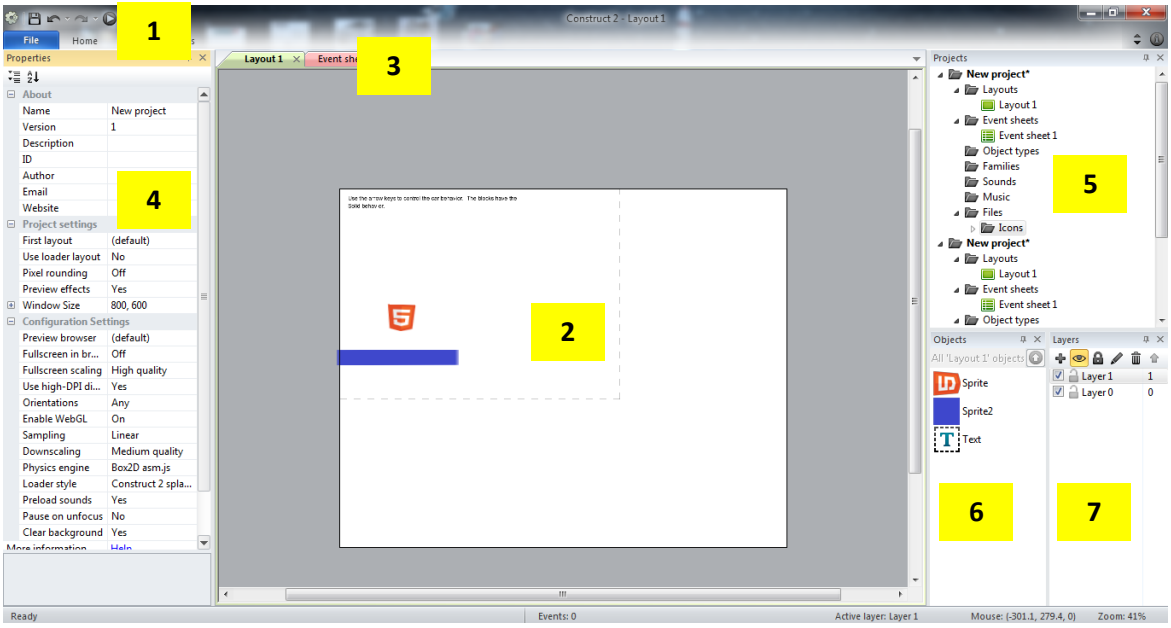
1. Mahasiswa terbiasa dengan interface Construct 2
2. Mahasiswa mengetahui kegunaan dari masing-masing behavior
3. Mahasiswa mengetahui cara penulisan parameter
4. Mahasiswa mengetahui kegunaan expressions serta cara penulisannya

DASAR TEORI

Construct 2 merupakan sebuah game engine berbasis HTML 5 buatan Scirra yang berasal dari London, Inggris. Dengan menggunakan game engine ini, programmer pemula maupun expert dapat dengan mudah membuat sebuah game. Construct 2 dikhususkan pada game berbasis 2(dua) dimensi yang menyediakan banyak fitur untuk mempercantik game yang ingin dirancang. Dalam Construct 2 tersedia 70 visual effect yang menggunakan engine webGL, serta dilengkapi dengan 20 built-in plugin dan behavior. Melalui Construct 2, programmer dapat mem-publish gamenya melalui beberapa platform seperti HTML 5, Google Chrome Webstore, Facebook, Phonegap, Windows Phone, Android, IOS, Tizen dan sebagainya.

Construct 2 adalah salah satu tools yang dapat digunakan untuk membuat game tanpa harus menulis kode pemrograman, karena sebagian besar logika untuk game dapat dibuat menggunakan menu. Kelebihan lain dari construct adalah fungsi-fungsi bawaan yang sudah disediakan dapat mempercepat proses pembuatan game, sehingga tidak perlu membuat ulang fungsi-fungsi tersebut untuk game yang akan di bangun.

1.1 Interface



Gambar 1.1 Interface Construct 2

1. Menu Bar & Ribbon Tabs

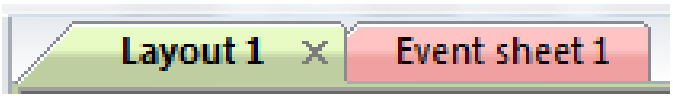
Gambar 1.2 Quick Access Toolbar

Tampilan menu dalam Construct 2 memakai bentuk ribbon. Tombol berbentuk roda gigi lambang Scirra akan menampilkan drop-down ribbon, yang di dalamnya berisi perintah-perintah standar seperti New, Open, dan Save (bentuknya berbeda-beda, tergantung pada versi engine). Di sebelahnya, terdapat quick access tollbar yang berisi empat buah perintah yang paling sering digunakan.

2. Layout

Layout adalah tempat bekerja. Anda bisa menempatkan objek, mendesain level, dan sebagainya. Tampilan layout dibagi menjadi dua, yaitu layout dan windows size. Layout adalah seluruh lembar kerja berwarna putih, sedangkan windows size mempresentasikan ukuran layar yang dipakai. Batas antara windows dan layout ditandai dengan adanya garis putus-putus. Ukuran layout dan windows size dapat diatur sesuai keinginan dengan melakukan konfigurasi pada **properties bar**.

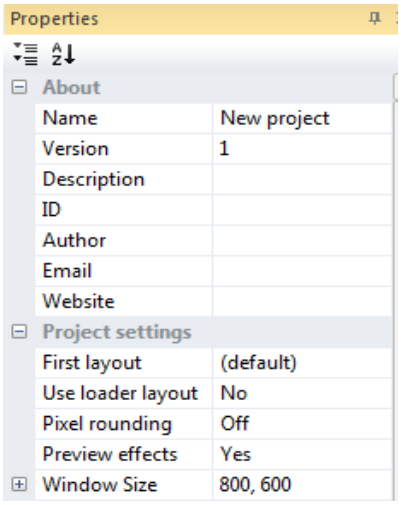
3. Tab



Gambar 1.3 Tabs

Tab berfungsi untuk mengganti layout maupun event sheet yang ingin dikerjakan. Anda dapat melakukan drag untuk mengatur urutannya. Layout yang aktif ditandai dengan munculnya icon close di bagian pojok kanannya. Untuk menutup tab, tinggal klik tanda close tersebut, sedangkan untuk membukanya kembali dapat dilakukan dengan mengaksesnya di folder **Layouts** dan **Event sheets** dalam **Project bar**.

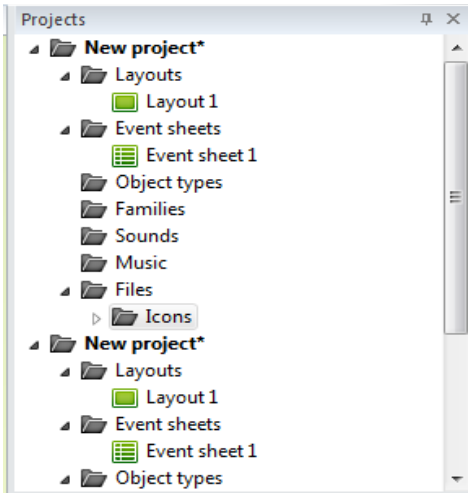
4. Properties Bar



Gambar 1.4 Properties Bar

Properties Bar berisi daftar pengaturan objek yang dapat anda ubah sesuai kebutuhan. Isi dari properties bar dapat berbeda-beda, tergantung pada objek apa yang dipilih. Opsi pengaturan ditujukkan dalam kolom kiri, sedangkan di kolom kanan dapat diisikan nilai sesuai yang diinginkan.

5. Project Bar



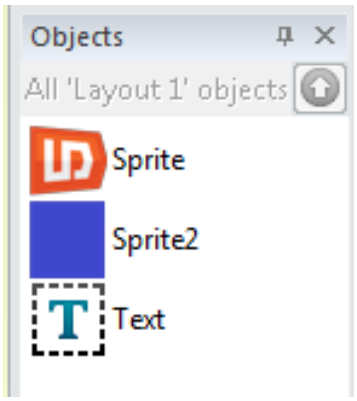
Gambar 1.5 Project Bar

Project Bar memberi gambaran umum tentang segala hal dalam proyek game yang dibuat, seperti jumlah event, layout, dan objek-objek yang dimiliki. Penambahan event sheet dapat dilakukan dengan cara **klik kanan pada Event sheets > Add event sheet.** Ketika project bar dipilih, maka project properties akan muncul pada properties bar. Berikut pengaturan yang sering digunakan :

- **First layout** : layout yang pertama kali akan dibuka ketika game dimainkan.
- **Windows size** : ukuran layar.
- **Preview browser** : memilih browser mana yang akan dipakai untuk melakukan playtest.,
- **Fullscreen in browser** : teknik-teknik penampilan layout dalam game.
- **Loader style** : mengganti logo saat proses loading.

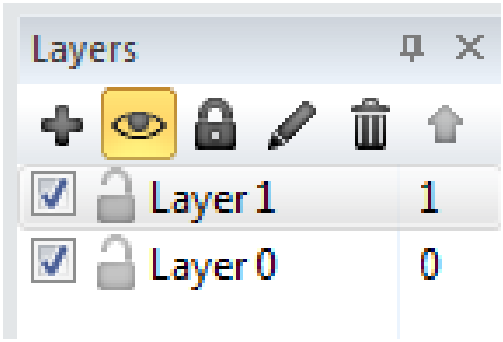
6. Object Bar

Object Bar berfungsi menunjukan objek secara spesifik, berdasarkan isi suatu folder dalam project bar. Drag and drop dapat dilakukan jika ingin memasukan objek ke dalam layout.



Gambar 1.6 Object Bar

7. **Layers Bar**



Gambar 1.7 Layers Bar

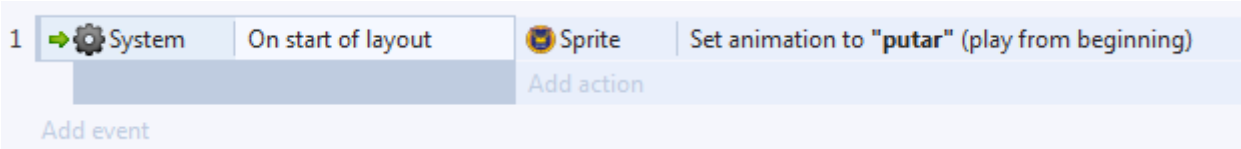
Layers bar digunakan untuk menambah, mengedit, maupun menghapus suatu layer dalam layout. Layer berperan besar untuk menciptakan kedalaman di game yang dibuat. Makin besar nilai layer, makin besar pula prioritas ojek dalam layout tersebut. Misalnya jika kita menaruh objek dalam layer UI, objek tersebut akan menutupi objek lainnya dalam layout yang lebih kecil nilainya.

Dalam menggunakan layer bar, perlu memperhatikan kegunaan tiap-tiap fungsi icon-nya. Icon berbentuk plus untuk menambah layer dan trash berfungsi untuk menghapus layer. Kemudian icon berbentuk mata berguna untuk men-disable layout hingga menjadi tak terlihat. Sedangkan icon bergambar gembok untuk mengunci objek-objek didalam layer untuk mencegah perubahan-perubahan yang tidak disengaja.

1.2 Events

Construct dapat mendefinisikan cara kerja game dengan memakai sistem blok logika, sehingga tidak memerlukan pengetahuan scripting maupun programming. Hal inilah yang membuat game engine ini mudah dan cepat dikuasai banyak orang. Proses eksekusi suatu event menggunakan logika sebab akibat atau jika-maka. Jika suatu kondisi dipenuhi, maka suatu perintah akan dijalankan.

Sub event adalah anak dari sebuah event lain. Ketika kondisi dari event induk terpenuhi, maka sub event juga akan dijalankan. Akan tetapi, walaupun kondisi sub event terpenuhi, namu kondisi event induk belum terpenuhi, maka perintah sub event tidak akan dijalankan. Perintah yang dijalankan sub event tidak bersamaan dengan main event, namun yang main event dulu yang dijalankan. Setelah semua perintah main event dilakukan, baru perintah dalam sub event dieksekusi.



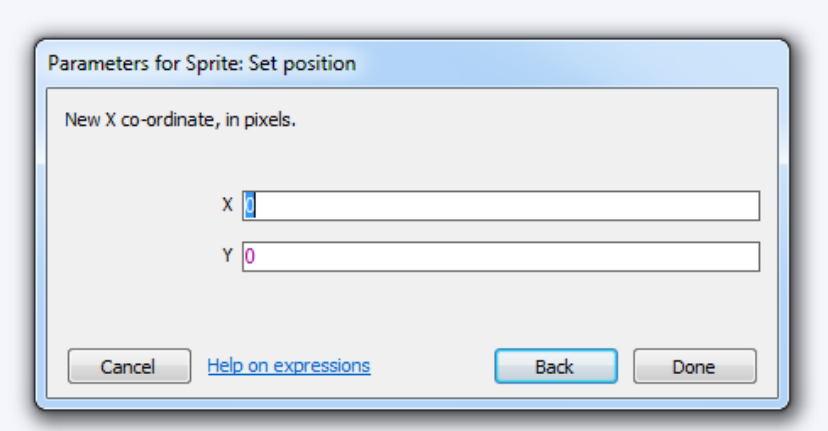
Gambar 1.8 Event

1. **Conditions:** syarat yang harus dipenuhi untuk melakukan suatu perintah.
2. **Actions:** kumpulan perintah yang dilakukan jika syaratnya sudah terpenuhi.
3. **Expressions:** berupa operasi logika maupun aritmetika. Bisa juga berisi nilai dari suatu objek atau variable.
4. **Sub events:** event yang berjalan ketika syarat dirinya dan event induk terpenuhi.
5. **Black sub event:** sub event yang tidak memiliki kondisi, sehingga apapun terjadi, aksi akan tetap diajalankan tanpa melihat kondisi sub event.
6. **Else:** kondisi yang bermakna "jika tidak" atau "selain itu". Misal pernyataan "jika hari minggu, maka sekolah libur", bentuk else-nya adalah "selain hari minggu, maka masuk sekolah".
7. **Groups:** berfungsi untuk mengelompokkan event berdasarkan perintah yang dijalankan.
8. **Comments:** untuk menulis catatan atau fungsi dari suatu event dengan menekan tombol "Q". Comment sangat penting jika untuk mengerjakan proyek game yang besar.
9. **Includes:** suatu event sheet di-include (dimasukan) ke event sheet yang lain, sehingga anda dapat memakai event sheet yang sama di layout yang berbeda tanpa melakukan copy-paste.
10. **Toogle disabled:** untuk mematikan sementara suatu event, sehingga menjadi tidak berfungsi walaupun kondisinya terpenuhi.
11. **Invert:** berfungsi membalik suatu pernyataan. Misal pernyataan "jika minum makan haus" setelah invert maka menjadi "jika tidak minum maka haus".
12. **Make 'Or' block:** beberapa kondisi yang berbeda dapat memiliki aksi yang sama. Untuk meletakan semua kondisi dalam satu event, maka digunakan "make 'Or' block". Contohnya, "jika player mati atau peluru habis, maka game over".
13. **Variables:** tempat menyimpan nilai suatu data. Sebuah variable dapat menyimpan nilai yang berubah-ubah, atau bisa juga konstan. Artinya, ia hanya dapat diakses, namun tidak dapat diubah nilainya.
    - **Global variables:** global variable terletak di event sheet paling atas. Kata "global" berarti variable ini dapat diakses dari mana saja, termasuk dari event sheet berbeda.

- **Local variables:** local variable hanya dimiliki oleh suatu group atau event. Local variables memiliki jangkauan tertentu, sehingga tidak semua event bisa mengaksesnya, walaupun dalam event sheet yang sama.
- **Text:** variable yang menyimpan karakter huruf atau angka. Karakter berupa angka ditandai dengan adanya tanda petik (“), misalnya “123”.
- **Number:** variable yang menyimpan angka, misalnya 456.
- **Initial value:** nilai awal dari suatu variable.

1.3 Object Paramenters & Expressions

Kotak dialog parameter akan muncul ketika menambahkan suatu aksi ke dalam event, bisa juga saat mengedit kondisi atau aksi. Namun, parameter hanya muncul saat menambahkan aksi tertentu saja, aksi seperti Destroy tidak membutuhkan parameter. Gambar dibawah ini merupakan salah satu contoh dialog penginputan parameter. Segala sesuatu yang di inputkan dalam parameter disebut **ekspresi (expressions).**

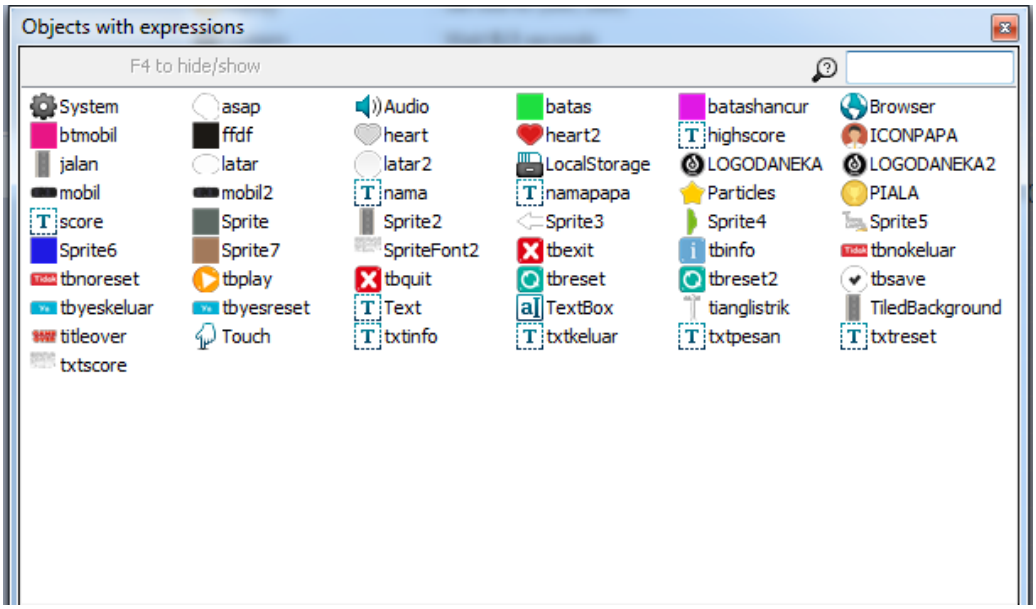


Gambar 1.9 Pengisian parameter untuk perintah Set Position

Expressions dapat berupa angka. Teks, atau suatu rumus. Ketika memasukkan ekspresi, akan keluar **Expresssions Panel** yang berisikan kumpulan ekspresi yang dimiliki oleh objek-objek dalam project. Terdapat deskripsi singkat mengenai fungsi dari setiap ekspresi, sehingga tidak akan membingungkan untuk mengetahui fungsinya.

Contoh ekspresi:

1. 15
2. “Hello World”
3. Player.Health + 50
4. “Highscore”&highscore
5. (distance(monster.X,hero.X,monster.Y,hero.Y))*(monster.Speed/5)



Gambar 1.10 Expressions panel dalam Tiang Listrik Hunter

Untuk mengambil nilai yang dimiliki oleh objek, digunakanlah object expressions untuk mendapatkannya. Caranya dengan mengetikkan nama_objek (dot) ekspresi, misal Mobil.Speed. Sebagai landasan untuk materi berikutnya, ada dua istilah penting yang perlu dipahami, yaitu objek dan instance. Sebuah objek dapat di ibaratkan adalah cetakan roti, sedangkan instance adalah roti yang dihasilkan. Satu buah cetakan dapat digunakan untuk membuat banyak roti. Begitu pula dengan objek, satu objek dapat digunakan untuk membuat banyak instance. Kesimpulannya, objek dan instance adalah dua hal yang berbeda.

Di dalam Construct 2, logika permainan diterapkan dengan blok-blok logika yang dinamakan event. Didalam event, ditaruhlah satu atau lebih kondisi, aksi hanya akan dilakukan pada instance yang memenuhi kondisi tersebut. Setiap instance berdiri sendiri, mereka tidak terhubung satu sama lain. Misalkan terdapat 5 buah instance monster. Ketika satu monster mati, maka monster yang lain tidak akan ikut mati.

Instance dapat dibuat saat runtime melalui event, atau juga bisa dibuat sebelumnya dalam layout untuk mendesain level, menu, dan sebagainya. Pada saat mendesain layout, Construct 2 mengharuskan membuat minimal satu buah instance dari setiap objek. Jika tidak, maka akan memunculkan pesan error.

1.4 Behavior Reference

Behavior merupakan fitur Construct 2 yang berfungsi membuat objek-objek di dalamnya memiliki perilaku tertentu. Berikut penjelasan mengenai behavior dan fungsi-fungsinya:

1. **8 Direction**
   Membuat object dapat digerakan dengan input tertentu. Arah gerakan objek bisa diatur sedemikian rupa, mulai dari dua, empat, hingga delapan arah.

2. **Anchor**
   Berfungsi untuk memposisikan objek secara otomatis agar sesuai dengan ukuran layar, hal ini berfungsi untuk mendukung berbagai ukuran layar.

3. **Bound to Layout**
   Berfungsi agar obyek tidak keluar dari layout game, sehinga seakan-akan ada tembok yang membatasi saat objek hendak keluar layout.

4. **Bullet**
   Berfungsi untuk membuat object maju lurus kedepan, ini biasa digunakan untuk peluru, tetapi bullet juga mempunyai opsi tambahan seperti gravitasi dan memantul yang digunakan untuk membuat object seperti bola yang memantul, selain biasa digunakan untuk peluru, bullet juga dapat digunakan untuk object sebagai musuh yang selalu bergerak secara otomatis

5. **Car**
   Berfungsi untuk membuat object dapat bergerak maju mundur belok kanan, kiri seperti memiliki kemudi, car biasanya digunakan untuk game yang bertema tentang kendaraan atau balapan

6. **Custom movement**
   Membuat obyek dapat bergerak sesuai kebiasaan (event based) movement.

7. **Destroy Outside Layout**

   Menghancurkan obyek setelah keluar dari layar utama game. Jika anda melihat peluru yang menghilang setelah keluar dari layar pada game, itu sebenarnya tidak menghilang, peluru itu akan tetap maju secara terus menurus dan jika hal ini dibiarkan lama kelamaan akan membuat loading game jadi berat. Untuk menghindari hal tersebut maka gunakanlah Destroy Outside Behavior yang akan menghancurkan object secara otomatis setelah keluar dari layar

8. **Drag And Drop**
   Berfungsi untuk memberikan sifat pada object agar dapat ditarik dan diposisikan sesuai keinginan dengan mengklik atau menyentuh obyek tersebut kemudian dapat dilepaskan jika posisi object sudah sesuai dengan yang anda inginkan dengan melepas klik atau sentuhan anda.

9. **Fade**
   Memberikan sifat pada object agar dapat memudar dan menghilang secara otomatis. Contohnya : jika anda menembak musuh dan tembakan tersebut mengenai musuh, maka akan keluar api dan api tersebut akan memudar dan menghilang secara otomatis.

10. **Flash**
    Membuat object dapat terlihat untuk beberapa saat lalu menghilang untuk beberapa saat kemudian muncul lagi sesuai waktu yang telah anda set dan akan terus berulang – ulang (seperti berkedip).

11. **Jump-Thru**
    Untuk membuat suatu pijakan dapat dipijak dan dapat ditembus dari bawah.

12. **Solid**
    Membuat suatu obyek dapat dipijak, sama seperti jump-thru. Namun, solid tidak dapat ditembus dari bawah.

13. **Line-of-Sight**
    Berfungsi untuk membatasi jarak pandang object. Seperti pada game peperangan, biasanya ada object yang menghalangi jarak pandang object pemain untuk melihat musuh. Misal terhalang tembok, pohon dan lain sebagainya

14. **No Save**
    Biasanya semua object dan tindakannnya akan disimpan dalam game, itu akan membuat loading game semakin lama semakin lambat. Dengan menggunakan no save behavior maka object yang telah dipasang no save behavior dan tindakan – tindakannya tidak akan disimpan dan tidak akan membuat loading game menjadi berat.

15. **Path Finding**
    Berfungsi untuk membuat object sebagai pemain dapat menemukan jalan tercepat disekitar rintangan secara cepat.

16. **Persist**
    Membuat object dapat mengingat tata letak yang berbeda pada saat ditinggalkan kemudian kembali lagi ke tempat tersebut. Object yang menggunakan persist behaviour disebut juga sebagai tata letak terus menerus. Ibaratnya, disaat anda telah menghancurkan dinding kemudian meninggalkannya, maka saat anda kembali lagi ke tempat tersebut kondisinya sama seperti saat anda tinggalkan (dindingnya tetap hancur).

17. Physics

Untuk contoh penggunaan physics behavior, anda lihat saja pada game Angry Bird dimana reruntuhan gedung berjatuhan kebawah dan jika salah satu object pada gedung yang roboh tersebut menyentuh object lain (gedung lain) maka object yang tersentuh akan ikut bergoyang atau bahkan ikut roboh.

18. Pin

Object yang diberi Pin Behavior akan memberikan kesan bahwa object tersebut telah disematkan atau menempel pada obyek lain.

19. Platform

Obyek yang diberi Platform Behavior berfungsi sebagai Pemain dalam game tersebut yang dapat digerakkan sesuai keinginan anda.

20. Rotate Behavior

Berfungsi agar game seolah-olah berputar.

21. Shadow Caster

Memberikan efek shadow (bayangan) pada object yang diberi Shadow Caster Behavior.

22. Sine

Dapat menyesuaikan object (seperti posisi, ukuran atau sudut). Seperti mebuat rumput bergoyang secara teratur dan terus menerus. Ini akan mempercantik tampilan game anda.

23. Timer

Berfungsi untuk memberikan batas waktu untuk pemain menyelesaikan permainan. Time Behavior digunakan hampir disetiap game.

24. Turret

Apakah anda pernah memainkan game contra? Jika pernah pasti anda melihat didalam game contra ada Tank yang dapat dinaiki dan mengikuti arah gerakan si object pemain. Nah, itulah fungsi dari Turret Behavior

25. Wrap

Ini berfungsi untuk me-repositions object. Misal pada permainan Snake II milik nokia, jika anda mengarahkan ularnya kebawah, maka setelah melewati batas ular tersebut akan muncul dari atas. Seperti itulah fungsi wrap.

26. Scroll To

Ini berfungsi untuk membuat kamera/view, seolah-olah mengikuti objek kemana pun berada.

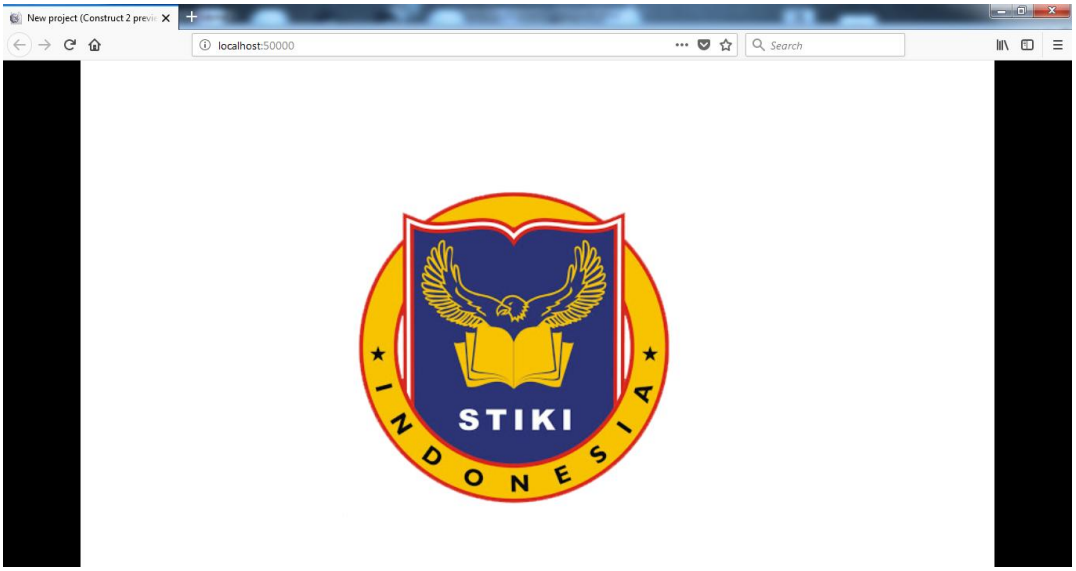
1.5 Playtest & Debug Test

Playtest & debug test adalah proses mencoba dan mencari kesalahan-kesalahan yang mungkin terjadi dalam game yang sedang dibuat. Playtest dapat dilakukan dengan menekan tombol "Play" dalam menu bar, sedangkan debug test dengan menekan tombol bergambar serangga.

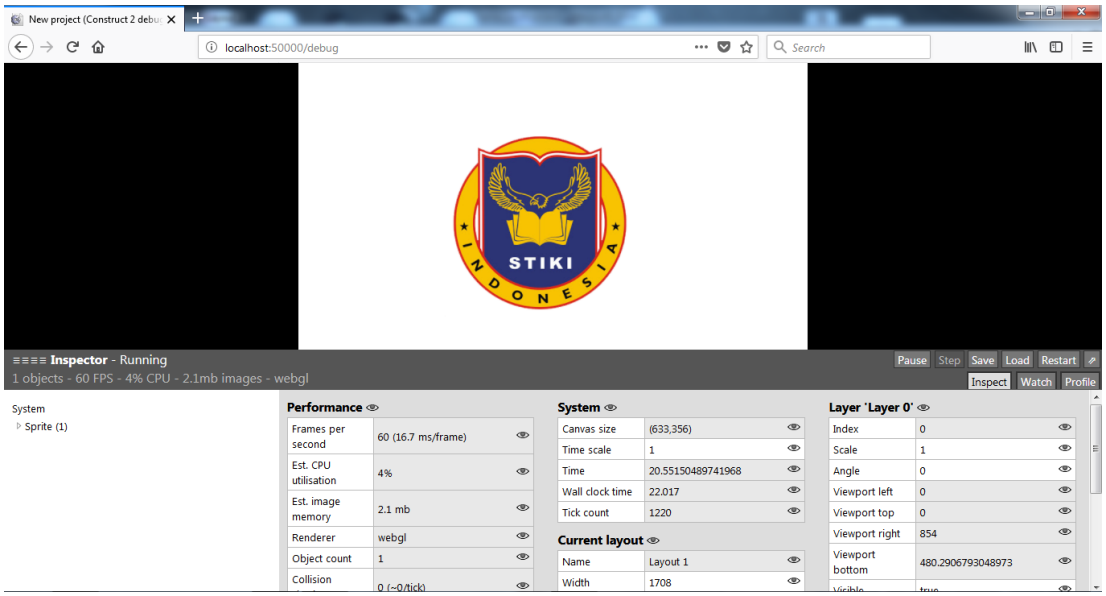
Gambar 1.11 Tombol Play dan Debug pada menu bar

Playtest dan debug test memiliki beberapa perbedaan. Playtest akan memberikan tampilan hasil akhir game yang sedang dibuat. Penilaian yang dilakukan cenderung ke arah estetika, seperti tata letak dan pemakaian warna. Jika ditemukan kesalahan, editing tidak dapat dilakukan dalam browser, namun dalam lembar kerja Construct. Kemudian jika sudah selesai, browser akan dibuka lagi, atau direload jika tidak ditutup.

Berbeda dengan playtest, penilaian yang dilakukan dalam debug cenderung ke arah teknis, seperti nilai variable, animasi yang sedang dimainkan, jumlah objek, dan lain-lain. Kemudian, tidak seperti playtest, dapat langsung dilakukan editing dalam bagian inspector jika terdapat kesalahan. Akan tetapi, fasilitas itu hanya untuk lisensi berbayar.



Gambar 1.12 Tampilan Playtest



Gambar 1.13 Tampilan Debug Test

TUGAS

1. Jelaskan dan sebutkan fungsi behavior-behavior yang ada di Construct !
2. Analisalah behavior apa saja yang digunakan game Super Mario !

MODUL II

INPUT OBJECT & PLAYER CONTROL

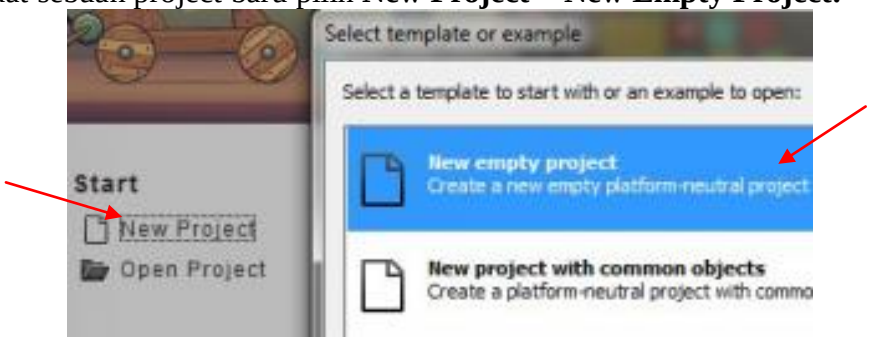
(Pertemuan 5)

Tujuan:

1. Mahasiswa memahami cara menambah objek dalam lembar kerja Construct 2
2. Mahasiswa memahami cara menggerakan karakter yang dimainkan

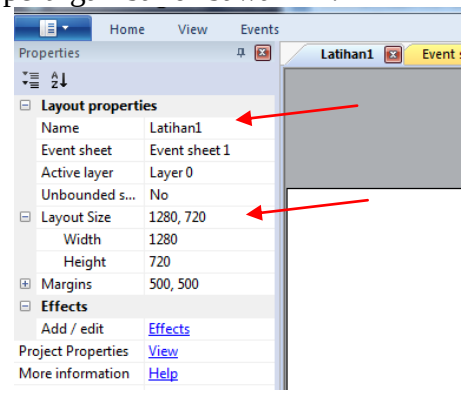
KEGIATAN PRAKTIKUM 2.1
MENAMBAH OBJEK PADA LAYOUT GAME

1. Pertama kali membuka Construct anda akan dihadapkan ke Start Page, disini anda membuat sebuah project baru pilih **New Project > New Empty Project.**



Gambar 2.1 Tampilan Halaman Awal Construct 2

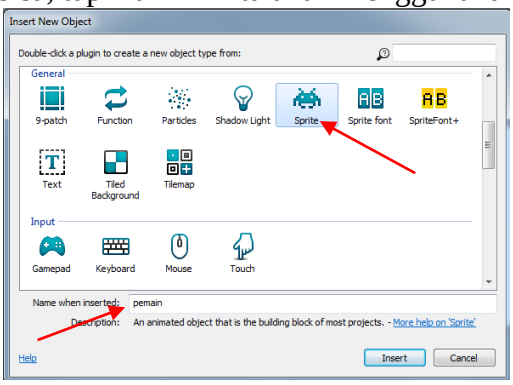
2. Perhatikan tabel di sisi kiri layar, itu adalah tabel properti. Anda bisa mengganti nama Layout pertama anda misalnya menjadi Latihan1, lalu pastikan anda ubah layout size menjadi (1280, 720) seperti gambar di bawah ini.



Gambar 2.2 Tabel properties

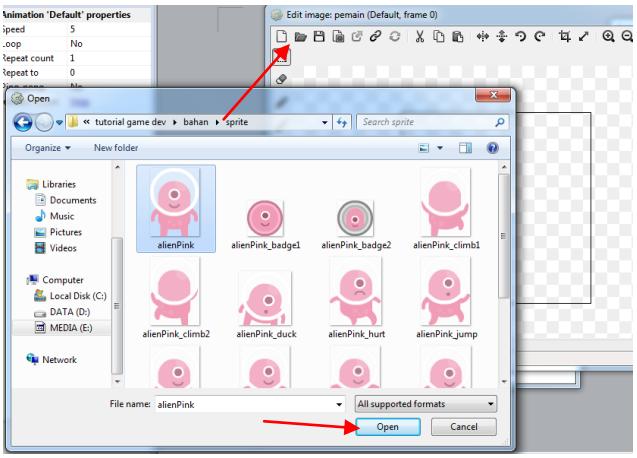
3. Untuk memasukkan gambar anda perlu klik **kanan > Insert new object > Sprite.** Sebelum klik insert jangan lupa biasakan memberi nama object yang kita masukan agar

mudah mengenali objek tersebut jika ingin mengolah objek tersebut dikemudian hari. Berilah nama objek tersebut misal "Pemain". Setelah itu anda akan diberi pilihan untuk membuat gambar baru atau mengambil gambar yang sudah ada. Silahkan pilih yang mana saja yang anda bisa, tapi kali ini kita akan menggunakan gambar yang sudah ada.



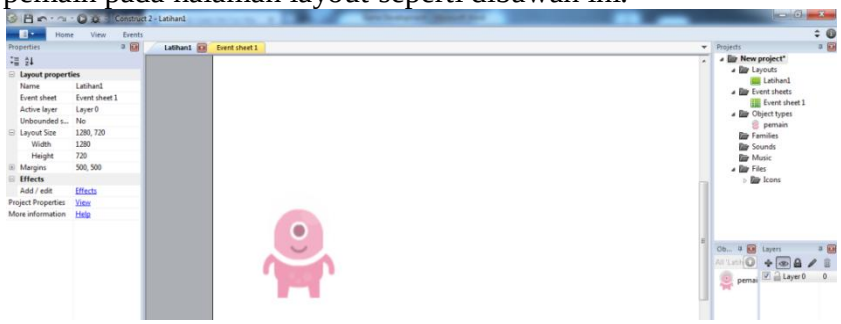
Gambar 2.3 Insert new object

4. Kemudian kita masukan karakter pemain yang ingin kita import dengan cara klik gambar icon folder pada edit image lalu pilih karakter yang ingin kita import sebagai pemain dalam game lalu klik open.



Gambar 2.4 Import sprite pemain

Setelah gambar sprite pemain sudah ter-import pilih close untuk melihat tampilan karakter pemain pada halaman layout seperti dibawah ini.

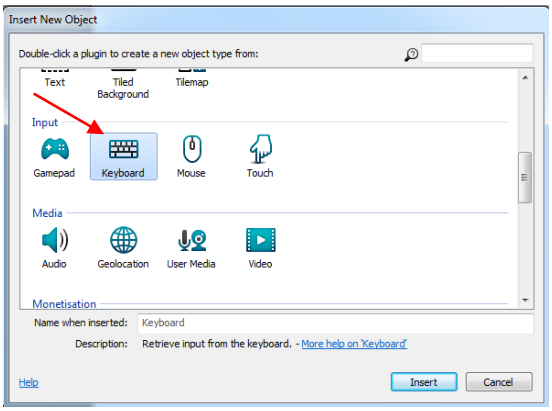


Gambar 2.5 Tampilan pemain pada layout

KEGIATAN PRAKTIKUM 2.2
MENGGERAKAN OBJEK KARAKTER GAME

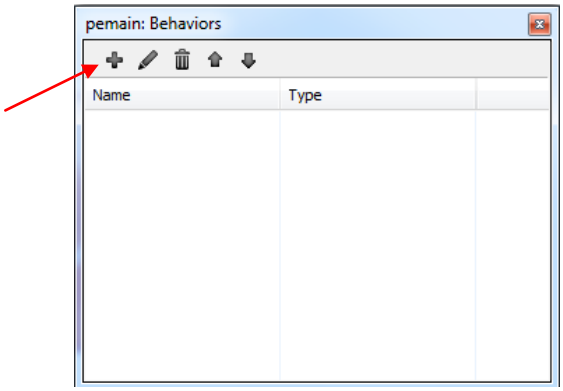
Construct 2 dapat mengenali berbagai macam input, bisa dari keyboard, gamepad, touchscreen, maupun dari mouse. Ada berbagai cara menggerakan objek, salah satunya bisa kita menggunakan behavior 8 Direction**.** Behavior 8 Direction ini membuat object dapat digerakan dengan input tertentu. Arah gerakan objek bisa diatur sedemikian rupa, mulai dari dua, empat, hingga delapan arah. Untuk lanjutan praktikum 2.1 kita akan menggunakan inputan keyboard dan menggunakan behavior 8 Direction.

1. Pertama kita akan memberikan plugin keyboard pada game yang akan dibuat agar sistem mengenali input dari keyboard. Caranya dengan klik kanan pada **layout > Insert new object > Keyboard.**



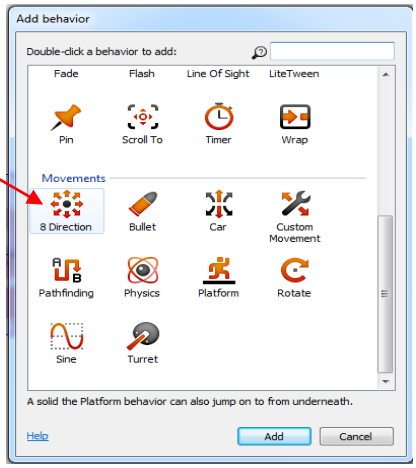
Gambar 2.6 Insert Input Keyboard

2. Setelah sistem mengetahui pemain akan memberikan input dari keyboard, selanjutnya kita memberikan behavior 8 Direction pada karakter pemain yang kita buat sebelumnya agar bisa bergerak, caranya dengan klik objek pemain, kemudian pada properties bar, cari sub-properties **behaviors > Add/edit Behaviors.**

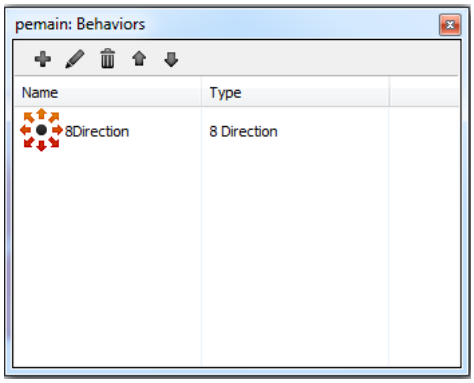


Gambar 2.7 Kotak Behaviors

3. Klik ikon tambah, kemudian akan muncul kotak dialog baru. Setelah itu, klik ikon 8 Direction.

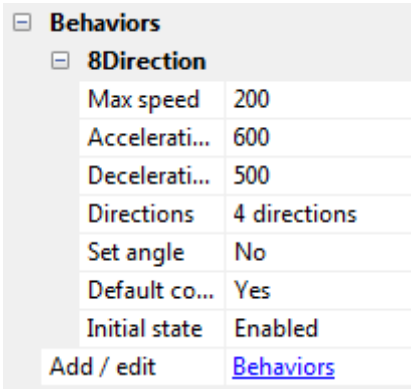


Gambar 2.8 Kotak dialog Add behavior



Gambar 2.9 Behavior yang sudah ditambah

4. Setelah berhasil menambah behavior 8 Direction, menu baru akan muncul dalam properties bar milik objek pemain.



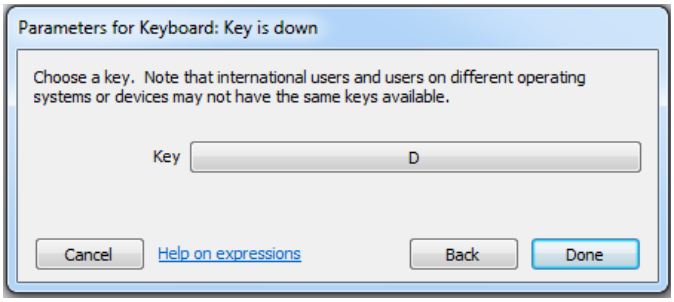
Gambar 2.10 Behavior objek pemain

5. Set pengaturan seperti gambar di atas, lalu lakukan playtest. Gerakan pemain dengan input tombol arrow pada keyboard.

6. Jika ingin menggunakan input lain selain tombol arrow membutuhkan logika bernama simulate control. Pertama alihkan tab ke:
   Event sheet1 > Add event > Keyboard > Key is down

Gambar 2.11 Memilih tombol keyboard sebagai input

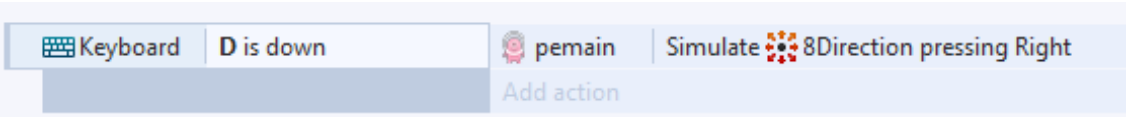
7. Klik tombol **<click to choose> > pilih tombol yang diinginkan.** Misal huruf "D" untuk menggerakan pemain ke kanan, lalu tekan **done.**



Gambar 2.12 Memilih tombol "D" sebagai input

Setelah menentukan kondisi, kita akan menentukan aksi yang akan dijalankan. Untuk menjalankan pemain ke kanan, lakukan langkah dibawah ini.

8. Buatlah eventsheet seperti dibawah ini



Gambar 2.13 Simulate control untuk tombol "D"

9. Ulangi langkah 6-8 untuk simulate control yang lain, dan pemain pun siap di gerakan sesuai arah yang di tentukan.

TUGAS

1. Dengan mengutak-atik properties bar, apa hasilnya jika behavior 8 Direction set angle-nya diganti 360 derajat ?
2. Apa yang terjadi saat Default Control dinonaktifkan ?

MODUL III

ANIMATIONS & CAMERA

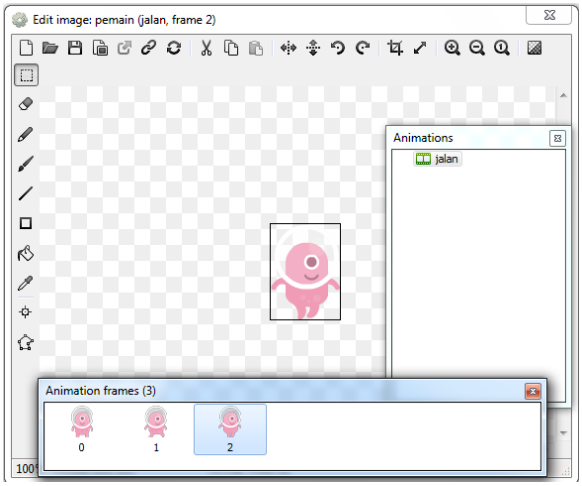
(Pertemuan 6)

Tujuan:

1. Mahasiswa memahami cara membuat animasi gerakan sesuai input yang di berikan pemain
2. Mahasiswa memahami teknik-teknik kamera dalam Construct 2

3.1 Animation

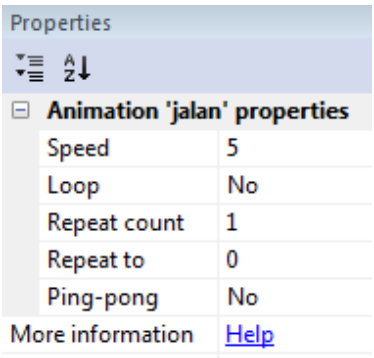
Salah satu implementasi dari animations adalah untuk membuat karakter atau objek lain untuk bergerak sesuai event yang diberikan. Untuk membuatnya anda perlu mengakses sprite editor dengan klik ganda pada sprite yang ingin diberikan animasi. Dalam sprite editor, animasi dibagi menjadi dua, animations dan animation frame. Animations berisikan kumpulan animasi yang akan dijalankan ketika suatu trigger aktif. Kita ambil contoh game Super Mario. Game tersebut memiliki beberapa animasi untuk Mario, di antaranya saat idle, berjalan, dan juga ketika mati. Ketika pemain memencet tombol kanan-kiri, maka sistem akan memainkan animasi berjalan ke kanan-kiri. Begitu pula jika pemain memencet tombol missal "X" untuk melompat. Ketika trigger di aktifkan, maka sistem akan memainkan animasi melompat.



Gambar 3.1 Kotak dialog sprite editor

Ketika mengedit sebuah sprite dalam sprite editor, menu yang ditampilkan dalam properties bar juga ikut berubah. Berikut penjelasan mengenai properties bar untuk sprite editor:

1. **Speed:** digunakan untuk mengatur seberapa cepat perpindahan frame. Makin besar nilainya, maka makin cepat perpindahannya. Namun jangan memberikan nilai yang terlalu besar, karena akan mengurangi performa untuk mobile devices.
2. **Loop:** ketika suatu animasi dimainkan pada suatu saat ia akan berhenti ketika suatu frame sudah dimainkan. Namun ketika loop diaktifkan, animasi akan dimainkan kembali dari awal ketika berhenti, begitu seterusnya sampai berulang-ulang tanpa berhenti.
3. **Repeat to:** misalkan anda memiliki lima frame, ketika animasi berhenti, maka sistem akan menayangkan frame terakhir, yaitu nomor lima. Dengan repeat to, anda dapat memilih frame nomor berapa yang ditayangkan ketika animasi berakhir.
4. **Ping-pong:** berfungsi untuk membalik urutan animasi yang dijalankan. Jika pertama kali menampilkan animasi dari depan ke belakang (misalkan 0-5), maka animasi kedua akan di tampilkan dari belakang ke depan (5-0).

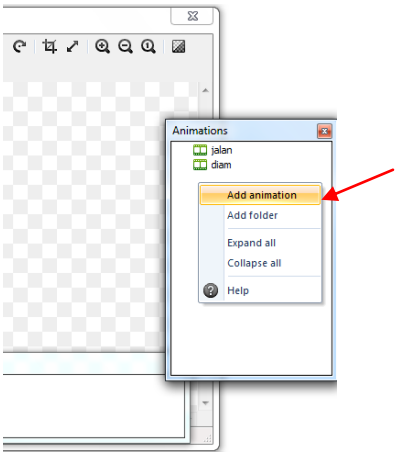


Gambar 3.2 Properties bar untuk sprite editor

KEGIATAN PRAKTIKUM 3.1
MENAMBAH ANIMASI

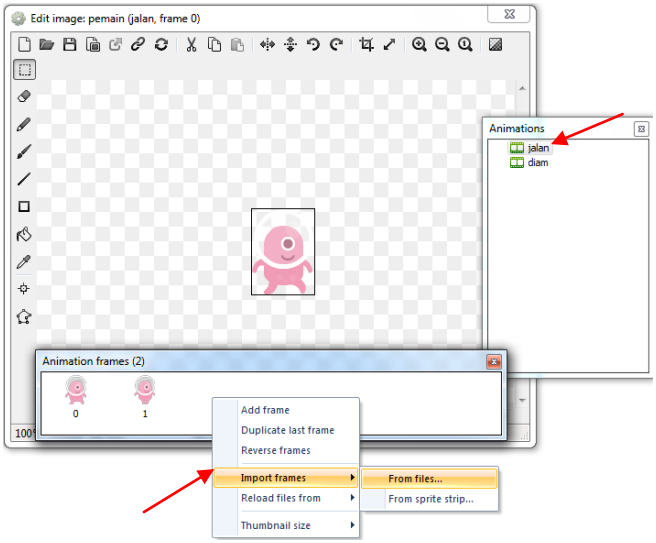
Pada praktikum 3.1 ini kita akan belajar mengimplementasi animation secara umum dengan membuka lagi file project yang kita buat pada modul II. Anda sudah membuat event untuk menggerakan pemain dari berbagai input. Pada praktikum 3.1 anda tinggal menambahkan animasi untuk membuatnya lebih realistis.

1. Klik ganda sprite pemain, maka akan muncul kotak dialog sprite editor. Pada bagian Animations, tambahkan dua buah animasi dan beri nama “diam” dan “jalan” dengan cara **klik kanan > Add animation**



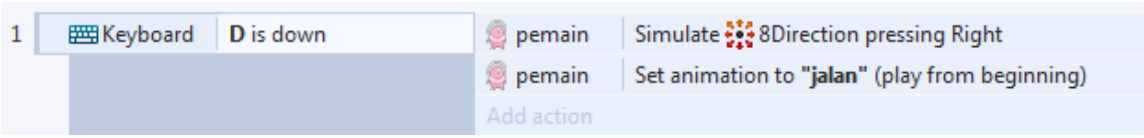
Gambar 3.3 Properties Bar Animations

Selanjutnya klik animasi jalan, dan klik kanan pada properties bar Animation Frame klik **Import frame > From files** untuk memasukan gambar yang sudah di siapkan untuk animasi jalan, lakukan hal yang sama pada animasi diam.



Gambar 3.4 Import gambar pada frames

2. Ganti tab ke event sheet 1. Di bawah ini simulate control untuk arah kanan, tambahkan satu aksi dibawahnya.

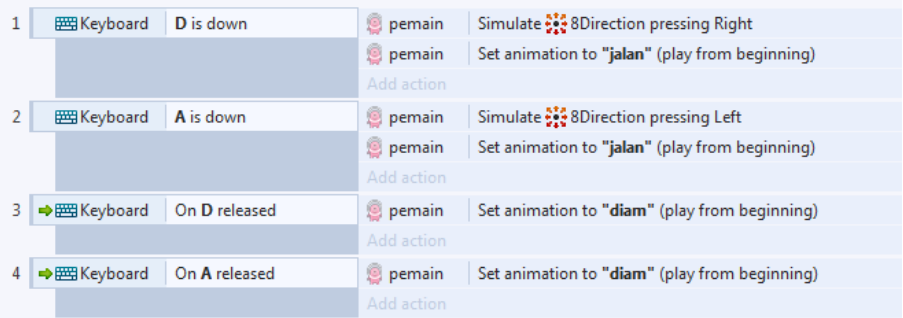


Gambar 3.5 Eventsheet untuk tombol "D"

Lakukan juga simulate control arah selanjutnya.

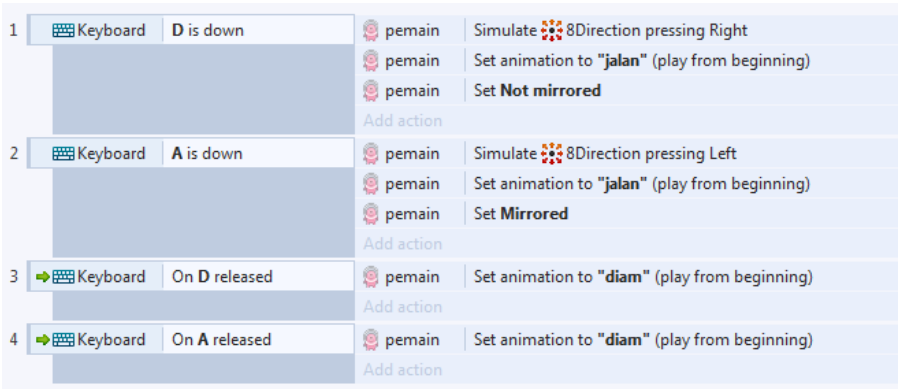
3. Lakukan playtest dan gerakan pemain. Animasi sudah berjalan, tetapi tetap playing walaupun kita sudah tidak lagi menekan arah kanan maupun kiri. Penyebabnya antara lain karena animasi yang dibuat hanya berjalan satu arah, yaitu dari animasi diam ke animasi jalan.

4. Untuk mengembalikan animasi ke state semula, ketika tombol A atau D tidak lagi di tekan, kembalikan lagi animasi ke diam. Tambahkan event sheet seperti dibawah ini.



Gambar 3.6 Eventsheet control animation

5. Jalankan lagi playtest dan lihat perubahannya. Animasi sudah bisa kembali ke state semula, dan di sini pemain saat berjalan ke kanan maupun ke kiri hanya menghadap satu arah, cara mengatasinya kita bisa menambahkan perintah Set Mirrored seperti di bawah ini agar pemain bisa menghadap berbalik arah sesuai tombol yang di tekan.



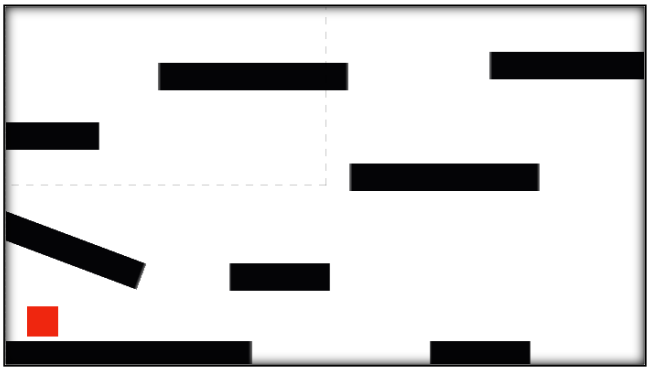
Gambar 3.7 Eventsheet Set Mirrored control animation

KEGIATAN PRAKTIKUM 3.2
MENAMBAH KAMERA DENGAN SCROLL TO BEHAVIOR

Pada praktikum 3.2 ini kita akan belajar membuat kamera mengikuti pemain kemanapun. Di dalam Construct 2 ada behavior yang bisa kita gunakan agar seolah-olah kamera mengikuti pemain yaitu Scroll to behavior.

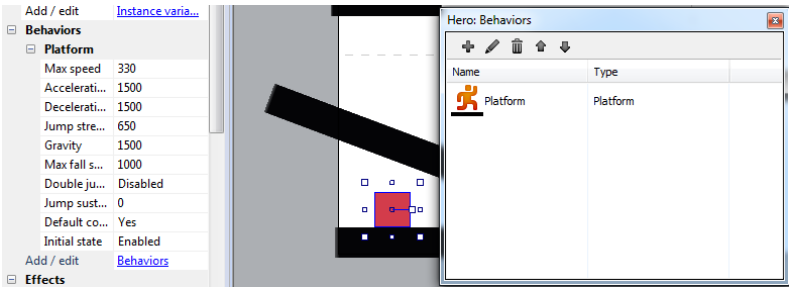
1. Cobalah mendesain sebuah level game bergenre platformer. Usahakan agar ukuran layout lebih besar dari ukuran windows size.

2. Beri nama karakter yang digunakan **"Hero".**

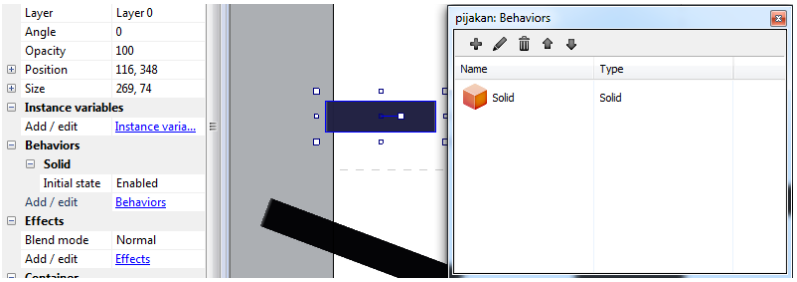


Gambar 3.8 Pastikan layout lebih besar dari windows size

3. Berikan **platform behavior** pada karakter, dan **solid behavior** atau **jump-thru behavior** pada platform yang digunakan sebagai tempat berpijak.



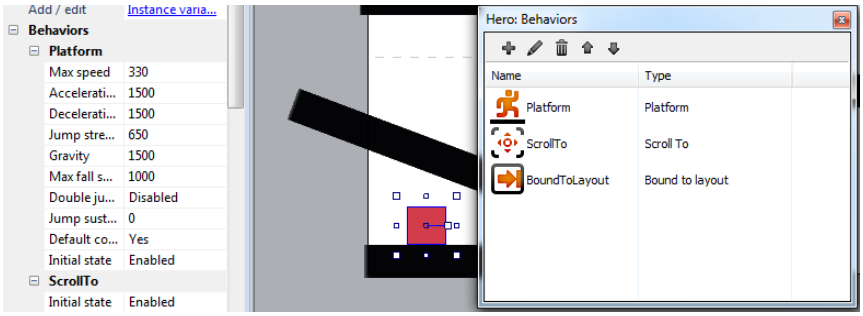
Gambar 3.9 Tambah Behavior Platform pada karakter Hero



Gambar 3.10 Tambah Behavior Solid pada pijakan

4. Gerakan karakter ke kanan, dan ia akan menghilang dari layar. Padahal level belum selesai dimainkan.

5. Untuk mengatasinya, berikan **scroll to behavior** pada karakter Hero agar kamera akan mengikuti kemana pun karakter pergi. Tambahkan juga **bound to layout behavior** pada karakter Hero, sehingga karakter anda tidak akan jatuh ketika berada di pinggir layout.



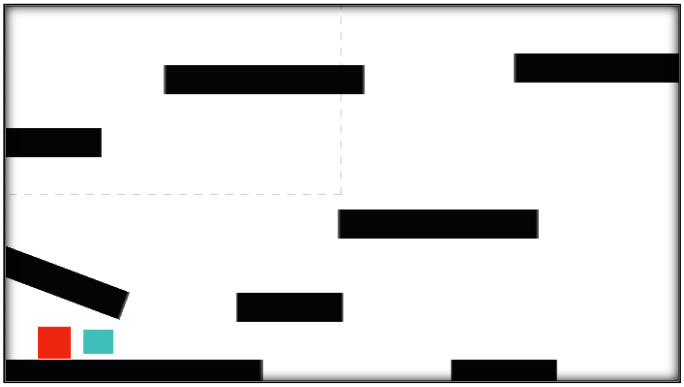
Gambar 3.11 Tambah Behavior Scroll to & Bound to layout pada karakter Hero

6. Jalankan playtest dan gerakan karakter maka kamera akan fokus ke karakter yang kita gerakan kemana pun bergerak.

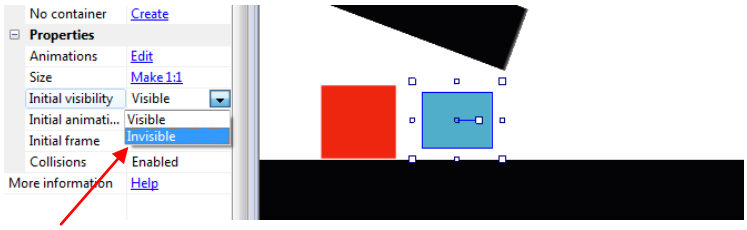
KEGIATAN PRAKTIKUM 3.3
LERP & DUMMY CAMERA

Pada praktikum sebelumnya kita sudah berhasil membuat kamera mengikuti pergerakan karakter, namun pergerakan kamera masih kasar. Hal itu karena sistem menempatkan karakter sebagai titik pusatnya (berada ditengah layar), sehingga sekecil apapun karakter bergerak, kamera akan memposisikan karakter anda di tengah-tengah layar. Oleh karena itu pada praktikum 3.3 ini kita akan perhalus gerakan kamera dengan ekspresi **lerp** dan **scroll.**

1. Buatlah satu sprite bernama **"Camera"** dengan **initial visibility=invisible** agar tidak terlihat ketika di mainkan, untuk pengaturan visibility berada di properties bar dengan cara klik sprite Camera. Berikan juga warna untuk mempermudah mencari posisinya di dalam layout.

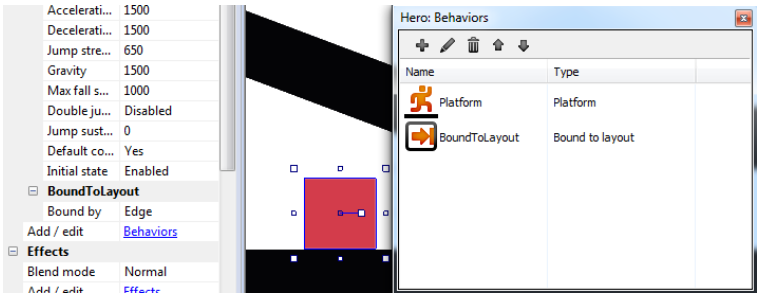


Gambar 3.11 Sprite "Camera" dengan warna biru

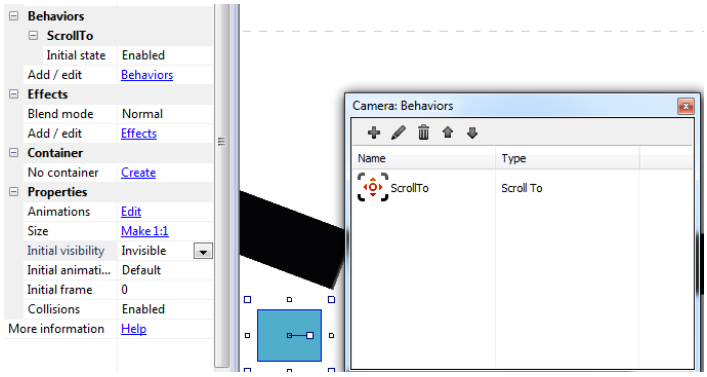


Gambar 3.12 Sprite "Camera" di invisible

2. Remove scroll to behavior yang kita buat pada karakter, dan sebaliknya, berikan behavior tersebut pada Sprite **Camera**.



Gambar 3.13 Remove scroll to behavior pada karakter Hero



Gambar 3.14 Tambah scroll to behavior pada sprite Camera

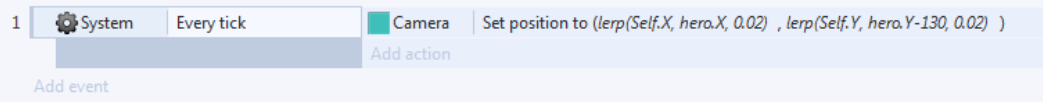
3. Buat event agar kamera mengikuti gerakan karakter seperti dibawah ini.

Add event > System > Every tick

Add action > Camera > Set position

X = lerp(Self.X, Hero.X, 0.02)

Y = lerp(Self.Y, Hero.Y-130, 0.02)



Gambar 3.15 Event sheet gerakan kamera

Anda memberi tiga buah nilai dalam ekspresi di atas. Pertama adalah posisi awal, kedua adalah posisi akhir, dan ketiga adalah waktu yang di butuhkan untuk bergerakn dari posisi awal ke akhir. **Self.X** dan **Self.Y** adalah posisi awal kamera, sedangkan **Hero.X** dan **Hero.Y** adala posisi dari player. Nilai **0.02** dapat diganti dengan angka berapa pun. Makin kecil angkanya, maka gerakan kamera makin lembut dan pelan.

4. Jalankan playtest dan gerakan karakter maka akan terlihat sangat halus gerakan kamera mengikuti posisi karakter Hero.

TUGAS

1. Buatlah game seperti super mario yang dimana berisi animasi berjalan, loncat, idle dan kamera mengikuti pemain kemanapun pergi.

MODUL IV

COLLISION & VARIABLE

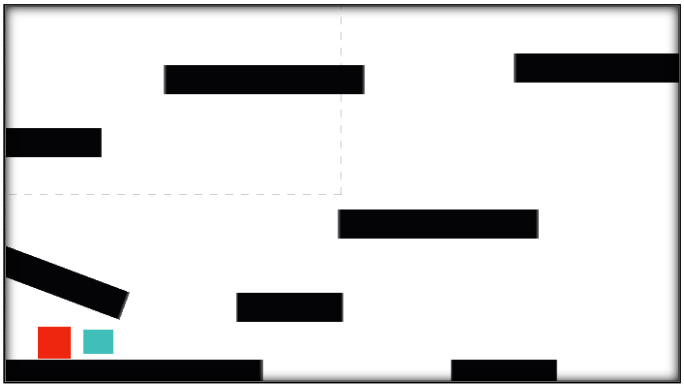
(Pertemuan 7)

Tujuan:

1. Mahasiswa memahami collision dalam Construct 2
2. Mahasiswa memahami variable dalam Construct 2

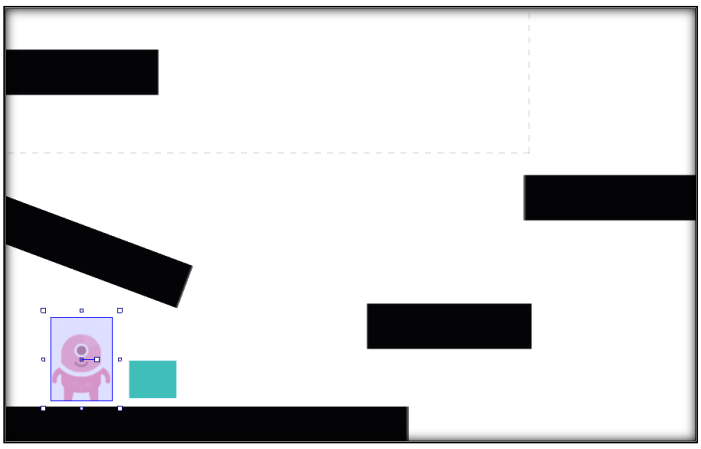
KEGIATAN PRAKTIKUM 4.1
COLLISION OBJECT

1. Pertama bukalah project yang sebelumnya kita buat di praktikum 3.3. Di pertemuan ini kita akan membahas collision object dalam Construct 2.



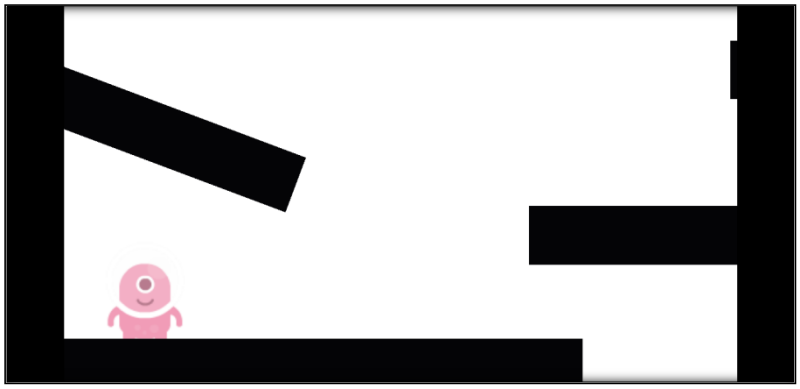
Gambar 4.1 Tampilan project praktikum sebelumnya

2. Klik ganda sprite karakter Hero, gantilah karakter tersebut dengan karakter yang berbentuk selain kotak.



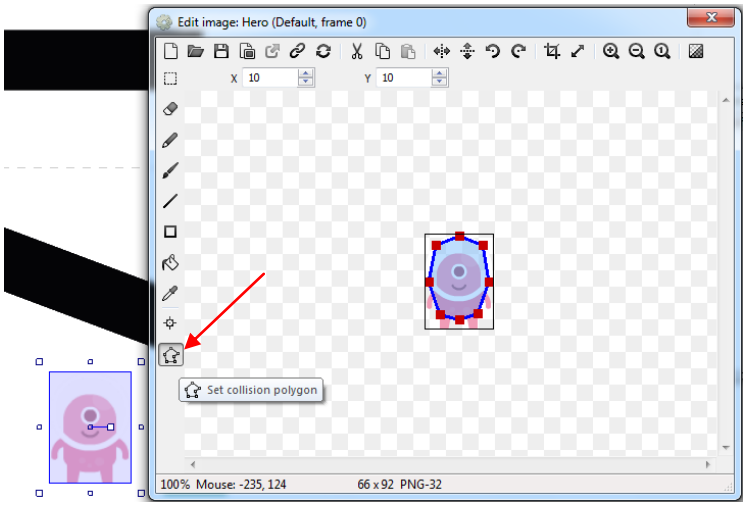
Gambar 4.2 Karakter Hero di ubah

3. Setelah mengganti karakter Hero cobalah playtest dan gerakan karakter apa yang terjadi apa karakter berjalan dengan sesuai? Disini karakter seakan-akan tenggelam setengah di pijakan.



Gambar 4.3 Tampilan karakter Hero yang tenggelam setengah di pijakan

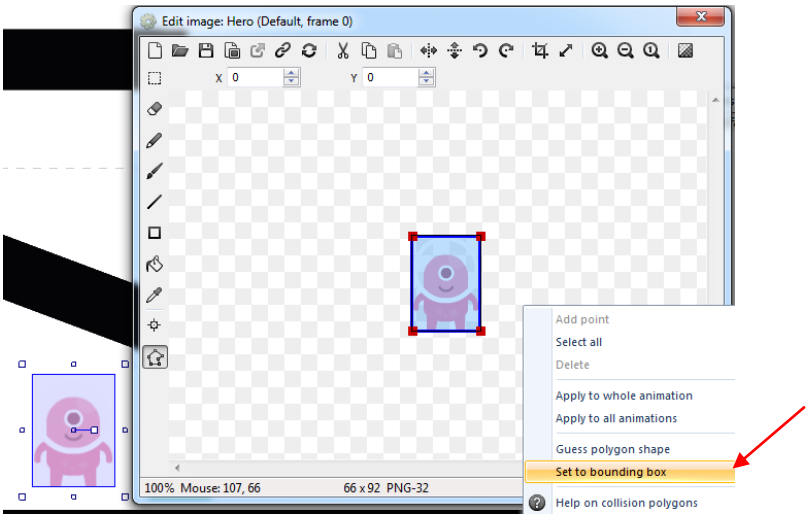
4. Cara mengatasi bug tersebut antara lain berhubungan dengan Collision yang kita bahas pada praktikum ini. Cobalah klik ganda karakter Hero dan klik set collision polygon.



Gambar 4.4 Tampilan collision polygon karakter Hero

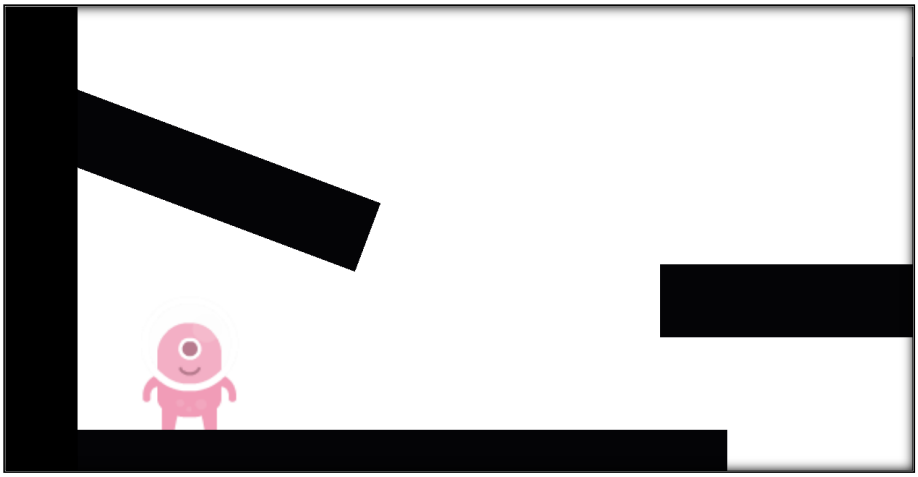
Disini bisa kita lihat bahwa collision karakter Hero hanya sampai selutut pemain, itulah penyebab karakter seakan-akan tenggelam kakinya di pijakan.

5. Disini kira bisa memperbaiki titik collision karakter Hero sesuai dengan posisinya, atau dengan cara cepatnya kita bisa menggunakan cara **klik kanan > Set to bounding box.** Dengan cara ini otomatis collision akan berbentuk kotak memenuhi karakter Hero.



Gambar 4.5 Tampilan collision karakter Hero yang di Set to bounding box

6. Setelah itu cobalah playtest dan lihat apakah karakter sudah sesuai dan sudah berjalan di atas pijakan. Disini terlihat karakter sudah berada di atas pijakannya dengan benar.



Gambar 4.6 Tampilan playtest karakter Hero

KEGIATAN PRAKTIKUM 4.2
VARIABEL UNTUK MENYIMPAN SKOR

Variabel adalah suatu mekanisme dalam pemrograman untuk menyimpan data yang bisa berubah saat program dijalankan. Contoh data yang berubah misalnya jumlah skor yang didapat, jumlah nyawa pemain, sisa waktu dalam permainan, dan lain-lain. Data yang bukan variabel dalam pemrograman ini disebut Constant atau konstanta.

Ada dua jenis Variabel dalam Construct 2:

1. **Variabel Instance** adalah bagian dari atribut sebuah objek, dan hanya ada selama objek yang bersangkutan ada di permainan. Jika karena suatu hal objek tersebut dihilangkan dari permainan, misalnya karakter lawan yang sudah dikalahkan pemain, Variabel Instance tadi sudah tidak bisa diakses lagi. Variabel jenis ini cocok digunakan untuk menyimpan informasi atribut suatu objek, misalnya untuk membedakan nyawa musuh (atau butuh berapa kali pukul untuk mengalahkan musuh tersebut).

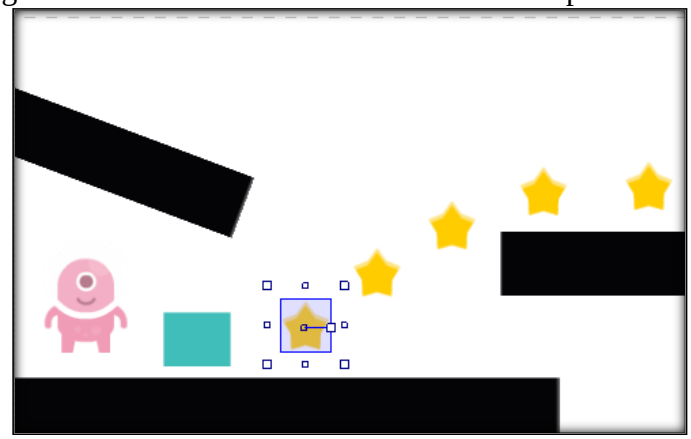
2. **Variabel Global** adalah variabel yang selalu ada selama pemainan dijalankan, karena tidak terikat pada objek. Variabel ini cocok digunakan untuk menyimpan informasi yang akan sering kita akses sepanjang permainan, misalnya skor, waktu, level saat ini, dan lain-lain.

Dalam praktikum kali ini kita akan membuat skor yang di dapat pemain menggunakan variabel.

1. Pertama bukalah project praktikum 4.1 yang kita buat sebelumnya

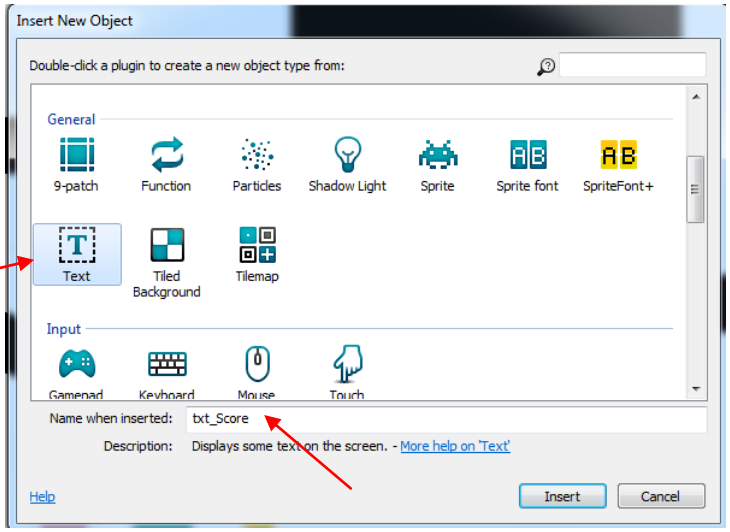
Gambar 4.7 Membuka project praktikum 4.1

2. Lalu kita tambahkan sprite “Bintang” seperti di bawah ini dimana sprite “Bintang” ini sebagai objek yang akan di tabrak karakter Hero untuk mendapatkan skor.



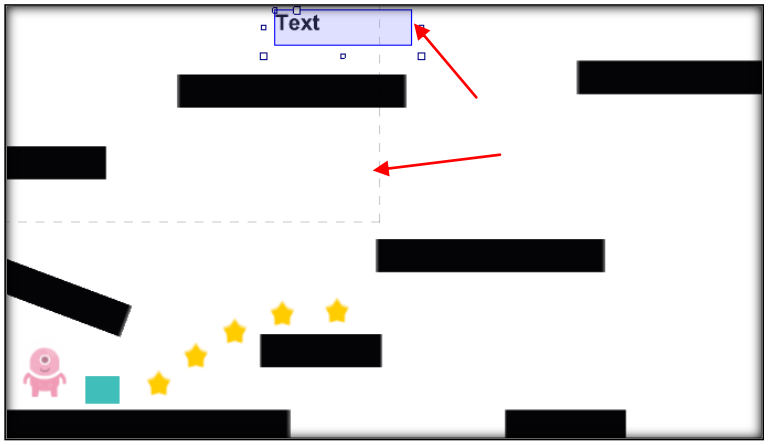
Gambar 4.8 Menambahkan Sprite Bintang di lembar kerja Construct 2

3. Selanjutnya tambahkan object Text untuk menampilkan Skor yang di dapat pemain dan beri nama **txt_Score**.



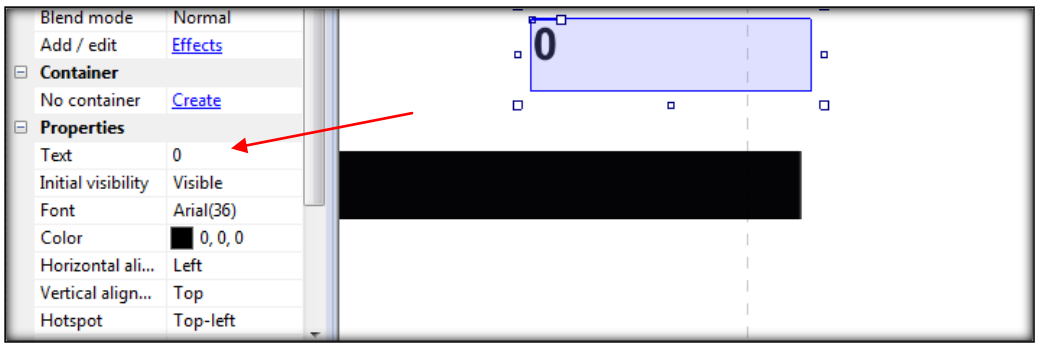
Gambar 4.9 Menambahkan oject Text untuk Skor pemain

Klik insert dan taruh txt_Score diposisi dalam windows size (lembar yang ada garis putus-putus) untuk menampilkan Skor pemain seperti di bawah ini.



Gambar 4.10 Menambahkan oject Text dalam windows size

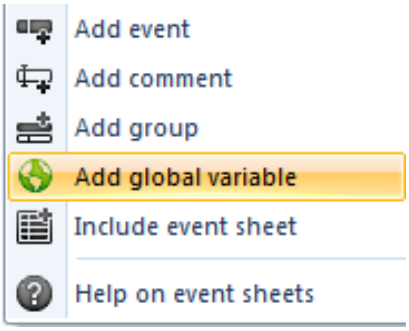
4. Ubah lah isi tulisan dalam txt_Score menjadi “0” melalu properties bar seperti dibawah ini.



Gambar 4.11 Mengubah isi txt_Score pada properties bar

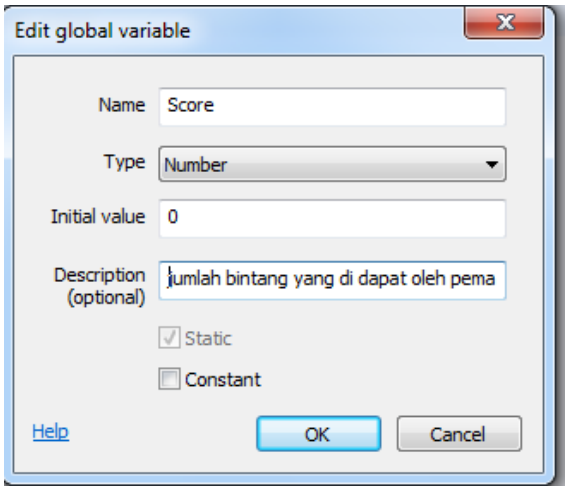
Untuk menyimpan skor kita perlu membuat suatu variabel global baru:

5. Klik kanan di halaman event > pilih "Add Global Variable".



Gambar 4.12 Add global variable

6. Beri nama yang mudah diingat dan menjelaskan isi dari variabel yang akan kita buat, misalnya "Score".



Gambar 4.13 Isian global variable

7. Pilih tipe variabel “Number” untuk menyimpan angka. Tipe Text hanya digunakan untuk menyimpan tulisan.

8. Untuk memperjelas kita bisa tambahkan keterangan “Jumlah bintang yang didapat oleh pemain”.

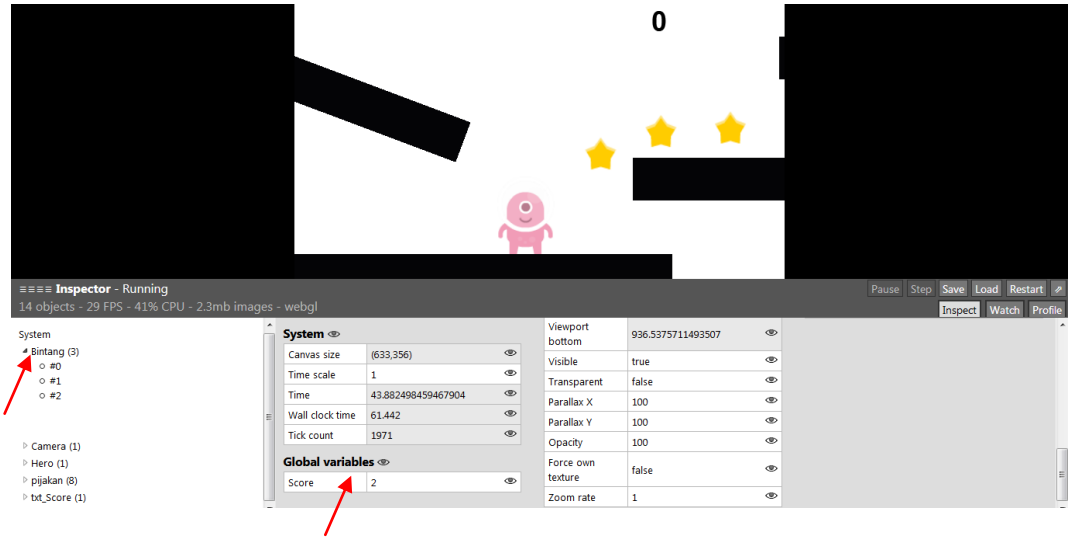
9. Biarkan opsi Constant tidak dipilih, karena kita akan merubah isi variabel ini nantinya. Jika dilakukan dengan benar, akan muncul variabel yang kita buat di bagian atas halaman event.

10. Setelah itu kita buat kondisi dimana saat pemain menabrak bintang maka bintang menghilang dan Score pemain bertambah dengan cara kita tambahkan evensheetnya seperti berikut

Gambar 4.14 Eventsheet menambah Score

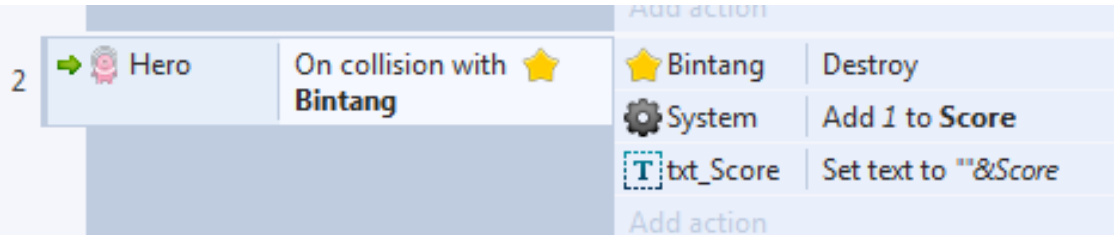
11. Setelah itu coba play dengan debug test dan lihat apakah Score sudah bertambah dan bintang menghilang saat ditabrak oleh pemain.

12. Di sini terlihat bintang sudah menghilang dari 5 bintang menjadi 3 bintang dan Score sudah bertambah menjadi 2, namun Score belum tampil dalam txt_Score sehingga txt_Score tetap “0”



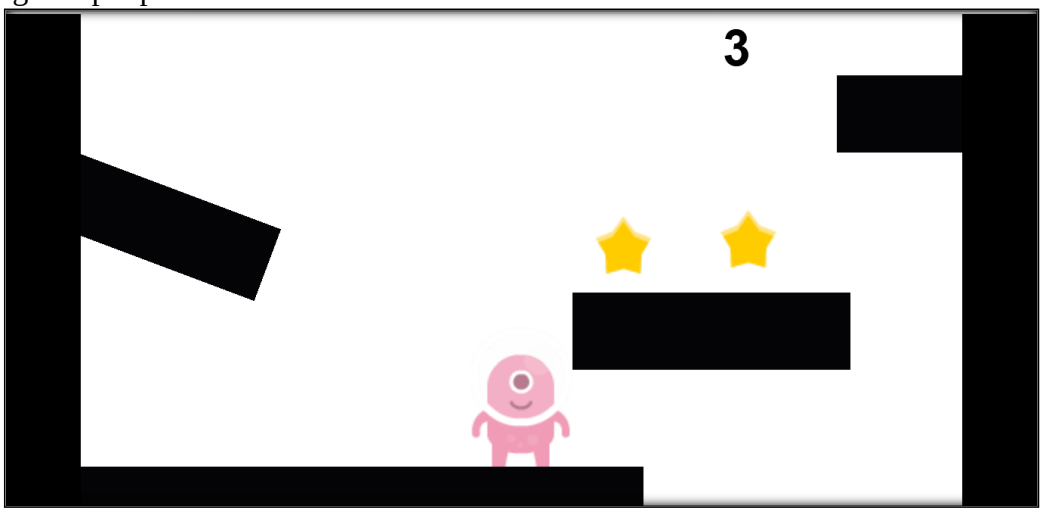
Gambar 4.15 Tampilan Debug Test Score

13. Agar nilai Score bisa tampil dalam txt_Score kita perlu menambahkan eventsheet dalam action yaitu Set Text agar Score bisa di baca oleh txt_Score seperti dibawah ini



Gambar 4.16 Eventsheet Set text txt_Score

14. Setelah itu coba Playtest dan lihat hasilnya, txt_Score berubah sesuai dengan score yang didapat pemain.



Gambar 4.17 Tampilan Playtest Score

TUGAS

1. Jelaskan apa itu variable ?
2. Dengan menggunakan variable pada game apa saja yang bisa di buat selain score ?
   .

MODUL V

HUD & FUNCTION

(Pertemuan 9)

Tujuan:

1. Mahasiswa memahami cara membuat HUD dalam Construct 2
2. Mahasiswa memahami Function dalam Construct 2

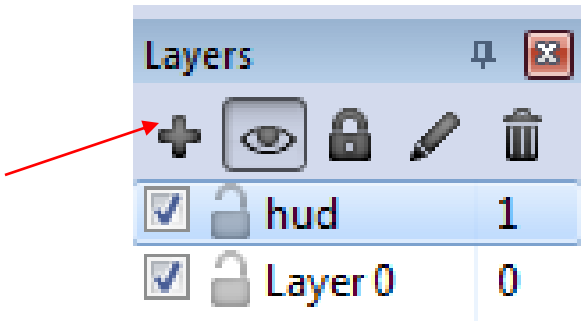
5.1 HUD (Head Up Display)

HUD (Head Up Display) adalah layar trasnparan yang memungkinkan melihat informasi tanpa perlu mengalihkan mata ke tempat lain. HUD menjadi bagian dari sistem informasi karakter game, misalnya kesehatan, skor, level, dan pemilihan senjata. HUD sangatlah penting dalam sebuah desain interface/antarmuka pada suatu game, yang dimana sangat mempengaruhi kenyamanan user dalam memainkan game.

KEGIATAN PRAKTIKUM 5.1
MEMBUAT HUD

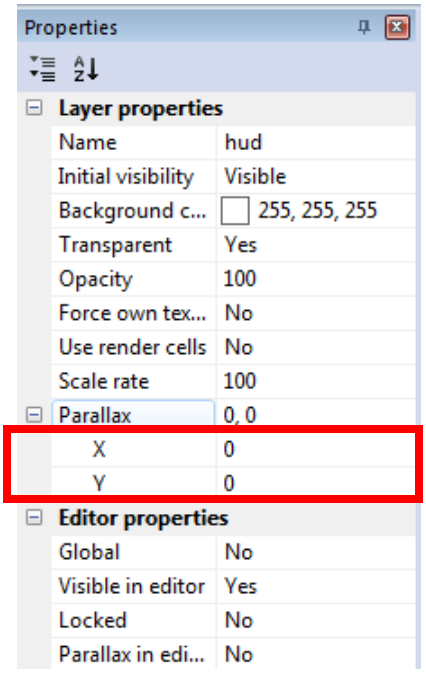
Pada modul V ini kita akan belajar mengimplementasi HUD dengan membuka lagi file project yang kita buat pada modul IV. Anda sudah membuat event untuk menambah score saat pemain saat menabrak bintang. Pada praktikum 5.1 anda tinggal menyeting agar txt Score yang kita buat pada modul sebelumnya tidak menghilang dari pandangan kita saat pemain berjalan dan tetap pada posisinya.

1. Klik Add layer pada layers bar untuk menambah layer baru, dan berikan nama HUD pada layer tersebut.



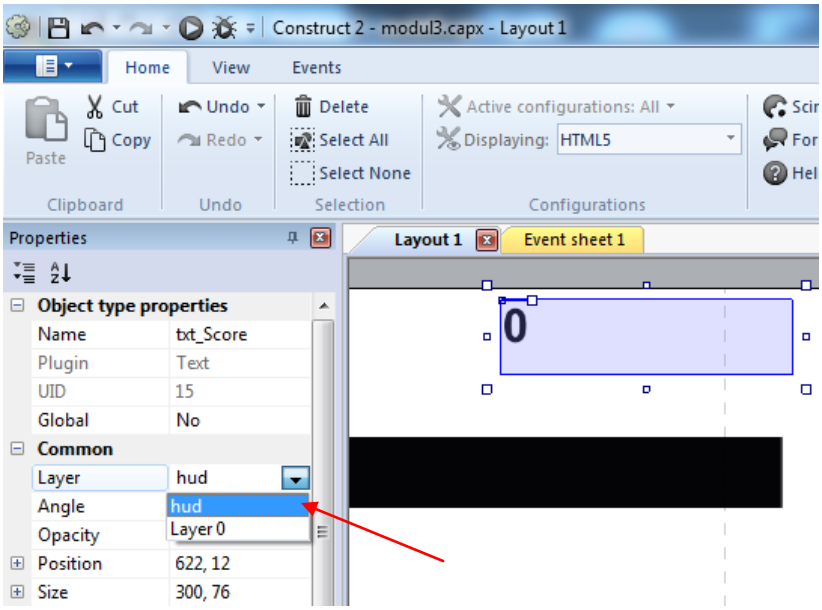
Gambar 5.1 Tambah layer HUD

2. Klik layer HUD dan lihat bagian layer properties, ubahlah nilai X dan Y pada Parallaxnya menjadi 0.



Gambar 5.2 Ubah nilai parallax layer HUD

3. Setelah itu klik txt_Score dan ubah layernya menjadi layer HUD pada properties.



Gambar 5.3 Ubah layer txt_Score

4. Lalu cobalah playtest dan lihat hasilnya, txt_Score akan tetap pada posisinya walau karakter berjalan.

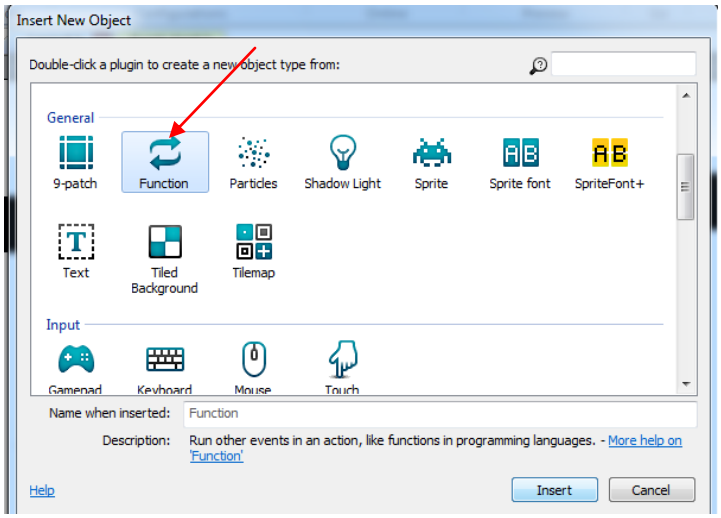
5.2 Function

Function adalah suatu kumpulan event yang melakukan aksi tertentu. Seringkali function dipanggil oleh bagian lain dari suatu program. Hal yang dikerjakan oleh function biasanya spesifik, sehinggga biasanya dipisahkan dari program utama. Keuntungan utama dari menggunakan function adalah menghemat ukuran program dan mengurangi duplikasi (penulisan kode yang sama). Misalnya, sebuah function menghitung menghitung luas dan keliling lima buah lingkaran yang berbeda jari-jarinya. Jika tanpa function, maka harus menuliskan kode satu persatu. Namun dengan function, cukup menuliskannya sekali, tinggal mengubah nilai jari-jarinya saja. Untuk memanggil function, maka lakukan aksi **Call function,** lalu untuk menjalankan function terkait menggunakan kondisi **On function.** Misalkan ingin menghitung luas dan keliling lingkaran maka kita membuat function untuk menghitungnya contohnya **On Function "Hitung"()**, maka kita tinggal memanggil function tersebut dengan **Call function"Hitung"()** sehingga akan lebih menghemat dalam menuliskan kode.Menggunakan function juga dapat meningkatkan kemampuan pencarian kesalahan selain itu juga dapat memecah event yang panjang menjadi lebih kecil.

KEGIATAN PRAKTIKUM 5.2
MEMBUAT FUNCTION

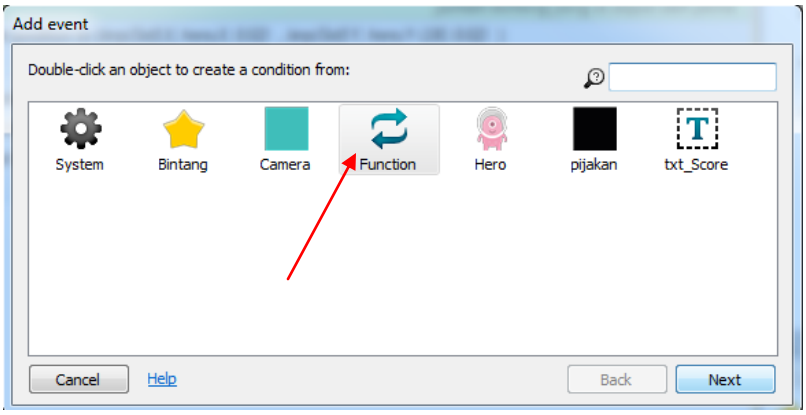
Pada praktikum 5.2 ini kita akan belajar membuat function yang akan membantu dalam menghemat penulisan kode dalam game yang dirancang. Sebelum memulai kita bisa membuka project yang kita buat pada praktikum 5.1.

1. Klik kanan lalu pilih Insert new object, cari object Function lalu klik insert dalam layout game yang kita buat.

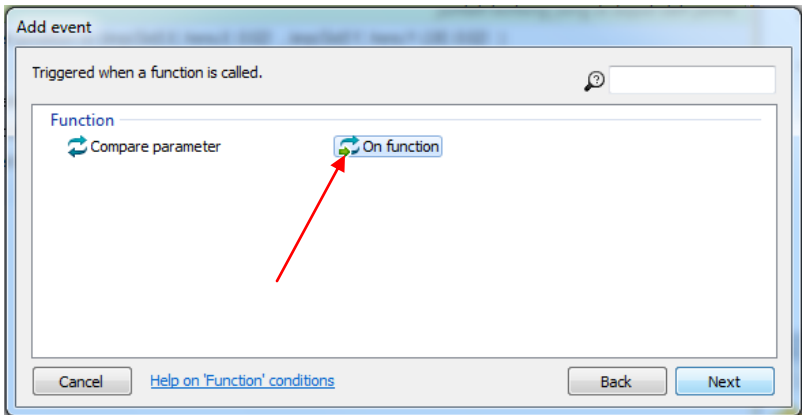


Gambar 5.4 Insert object function

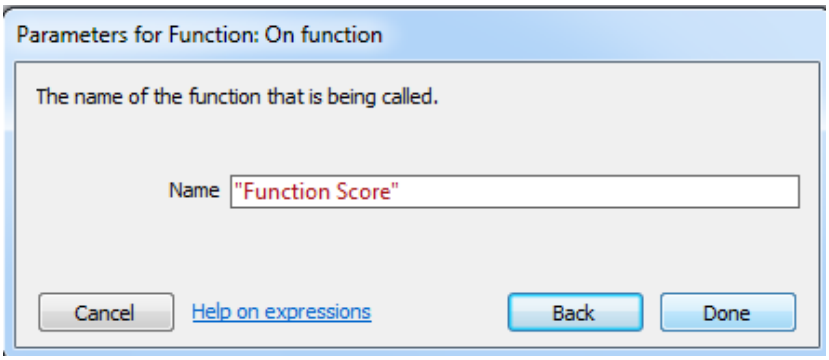
2. Setelah object function di insert kita pergi ke eventsheet untuk membuat function, dengan cara klik Add event > Function > On function > ketikan nama function "Function Score".



Gambar 5.5 Tambah Function langkah ke-1

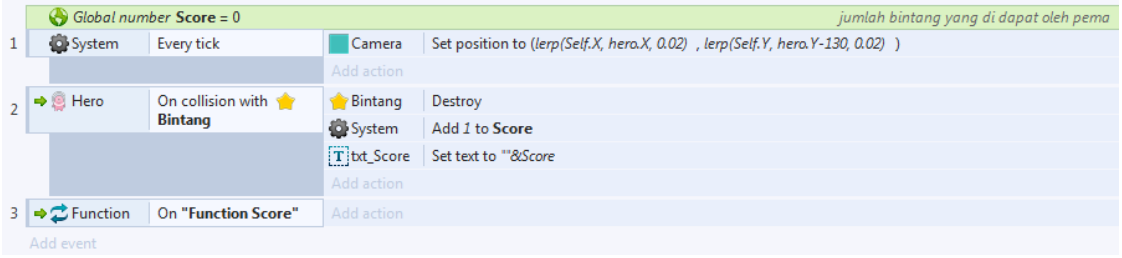


Gambar 5.6 Tambah Function langkah ke-2



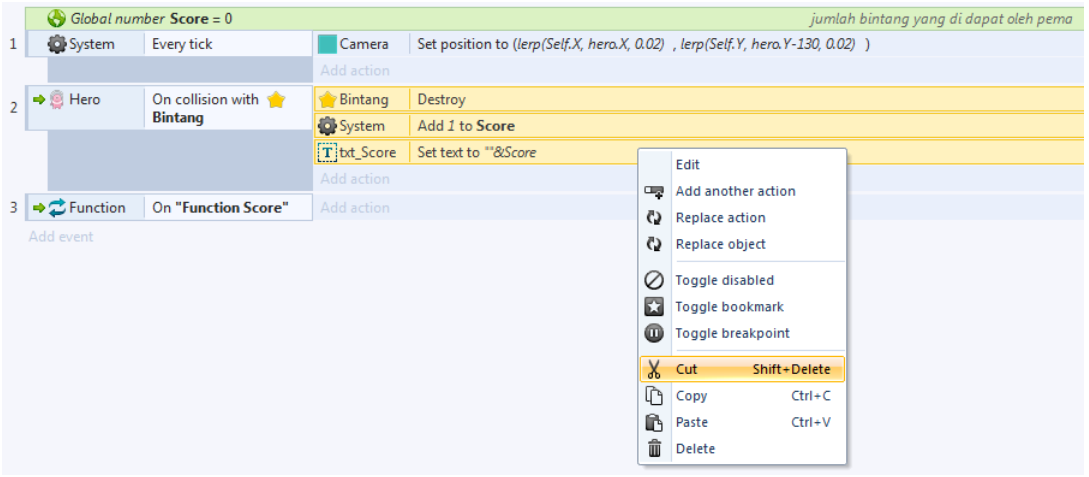
Gambar 5.7 Tambah Function langkah ke-3

Setelah mengikuti langkah diatas maka akan eventsheet akan seperti berikut

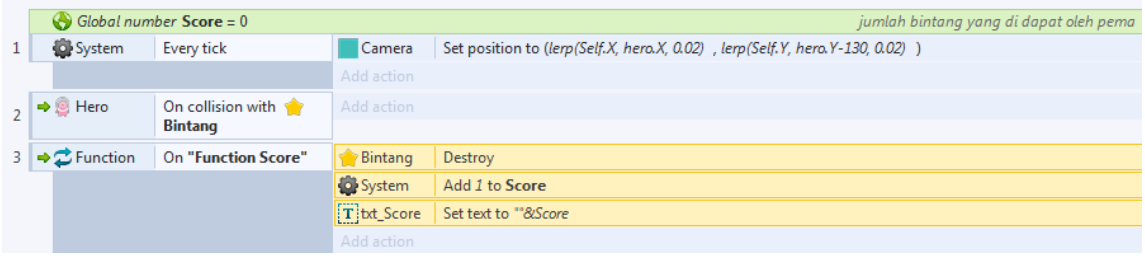


Gambar 5.8 Tampilan eventsheet tambah function

3. Setelah langkah-langkah diatas dilakukan kita cut isi action pada baris kedua saat pemain menabrak bintang dan paste di baris action function yang kita buat sebelumnya seperti dibawah ini.

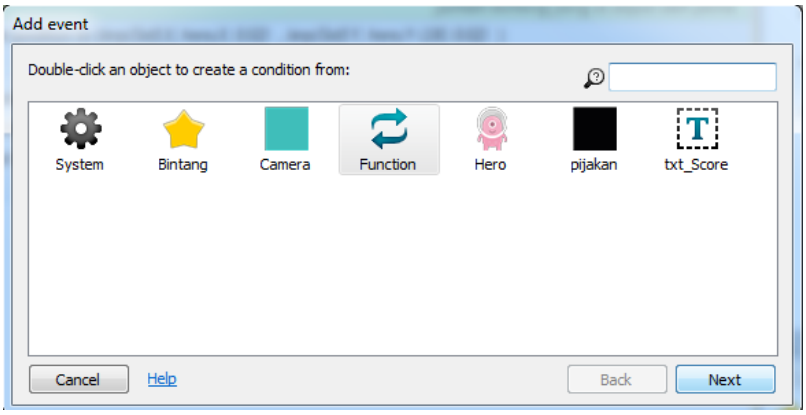


Gambar 5.9 Pindah isi action baris ke-2 ke function langkah ke-1

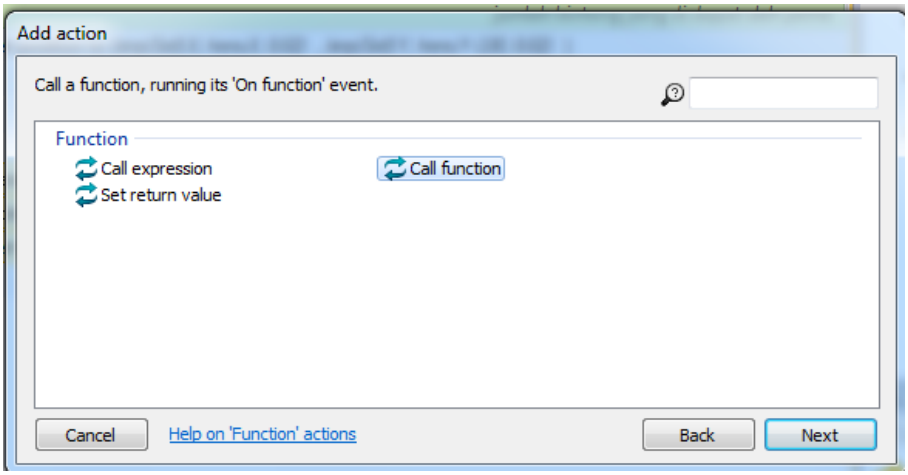


Gambar 5.10 Pindah isi action baris ke-2 ke function langkah ke-2

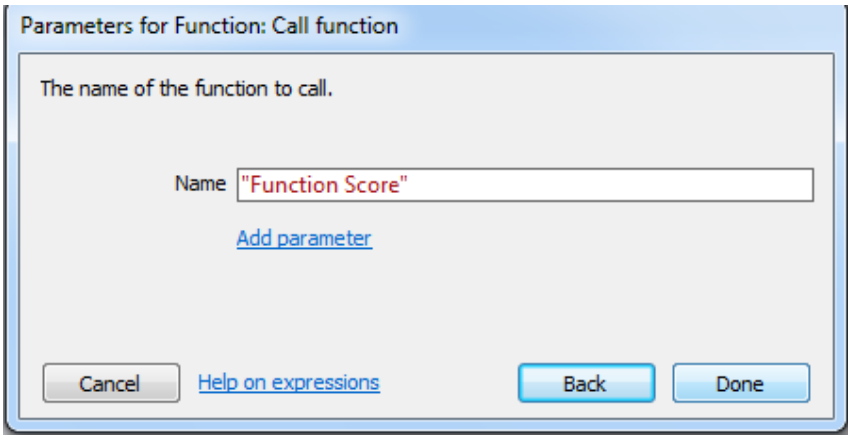
4. Lalu klik action pada baris ke-2 Add Action > Function > Call Function > Ketikan nama function yang akan dipanggil “Function Score”.



Gambar 5.11 Panggil Function langkah ke-1



Gambar 5.12 Panggil Function langkah ke-2



Gambar 5.13 Panggil Function langkah ke-3

Setelah mengikuti langkah diatas maka akan eventsheet akan seperti berikut

Gambar 5.14 Tampilan eventsheet panggil function

5. Lalu coba playtest dan lihat hasilnya, game tetap berjalan seperti biasanya, pada praktikum ini kita telah berhasil membuat function sederhana dan function ini sangat cocok di kembangkan untuk kode-kode yang sering dilakukan berulang kali sehingga tidak menghabiskan banyak baris kode.

TUGAS

1. Jelaskan apa itu HUD dan sebutkan contoh HUD pada game !
2. Jelaskan apa itu Function dan apa keuntungan dalam membuat function !

MODUL VI

EXPORT GAME HTML 5

(Pertemuan 10)

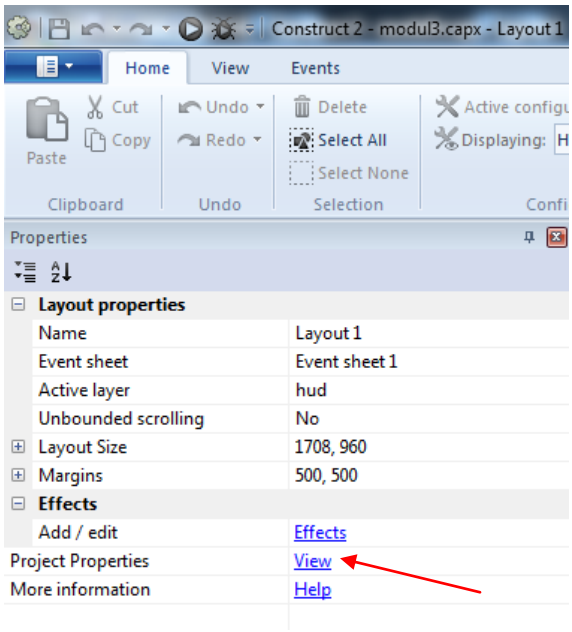
Tujuan:

1. Mahasiswa memahami cara mengexport game dari construct 2 ke HTML 5

KEGIATAN PRAKTIKUM 6.1
EXPORT GAME HTML 5

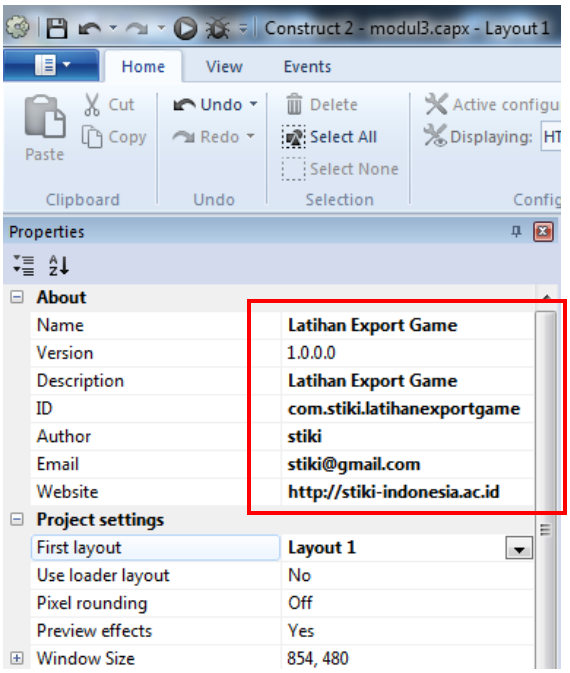
Pada praktikum 6.1 ini kita akan belajar cara mengexport game yang sudah dibuat sebelumnya ke dalam bentuk HTML 5.

1. Pertama bukalah project game yang akan di export.
2. Klik view pada properties bar



Gambar 6.1 Export game langkah ke-1

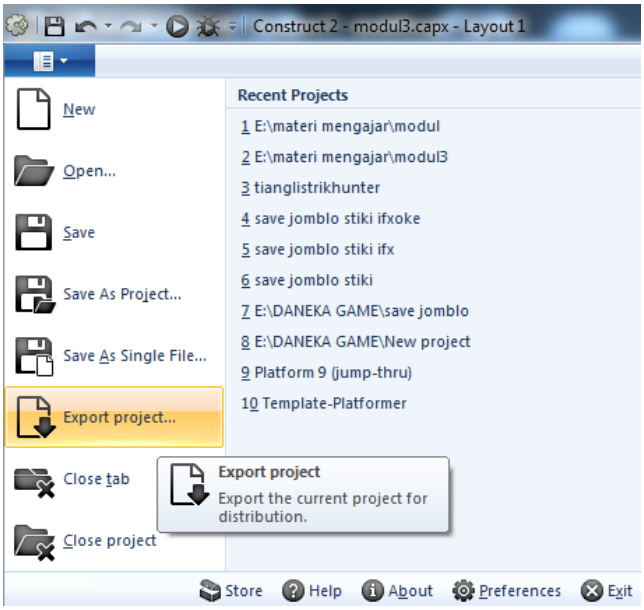
3. Isilah data about pada properties dengan lengkap seperti dibawah ini jangan sampai ada yang kosong. Pada bagian ID di harapkan defaultnya wajib diubah contoh "com.stiki.latihanexportgame".



Gambar 6.2 Export game langkah ke-2

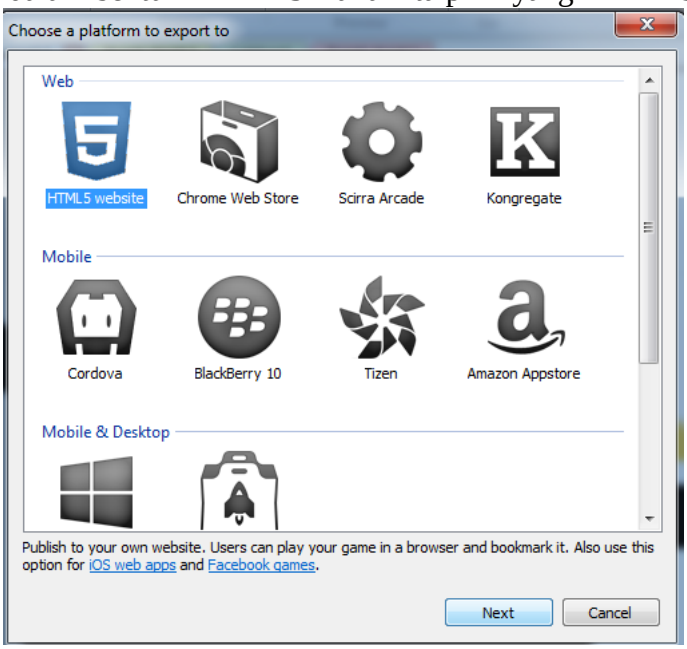
Pada bagian First Layout diharapkan di set sesuai layout apa yang akan kita play saat game mulai contoh kita set "Layout 1" sebagai layout awal yang kita play.

4. Setelah itu klik File > Export project



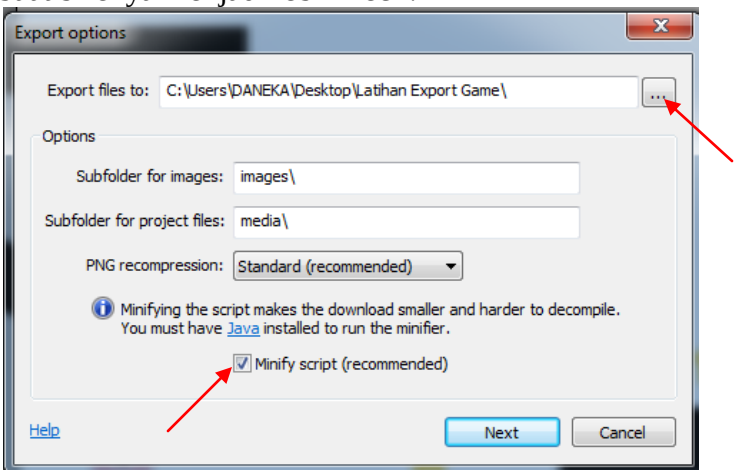
Gambar 6.3 Export game langkah ke-3

5. Pada tahap ini kita akan melihat banyak pilihan untuk export game, karena kita ingin export game kita dalam bentuk HTML 5 maka kita pilih yang HTML 5.



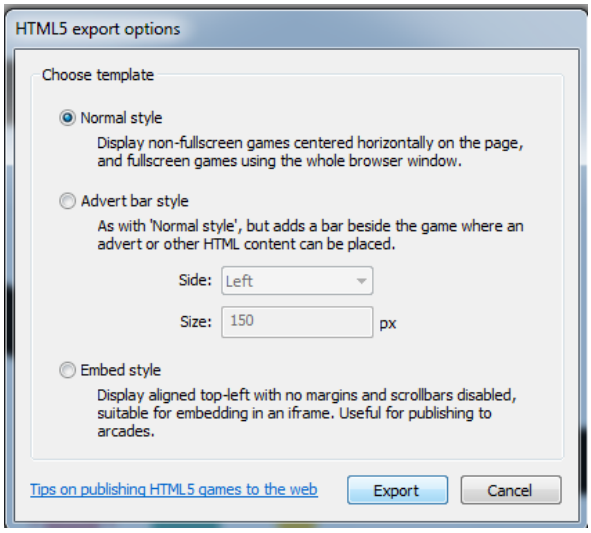
Gambar 6.4 Export game langkah ke-4

6. Dalam Export files to aturlah dimana game kita akan diexport dan bagian manify script boleh di centang boleh tidak, jika pada komputer tidak ada java maka manify script tidak perlu di centang. Namun manify script sangat berguna untuk membuat game yang kita buat sizenya menjadi lebih kecil.



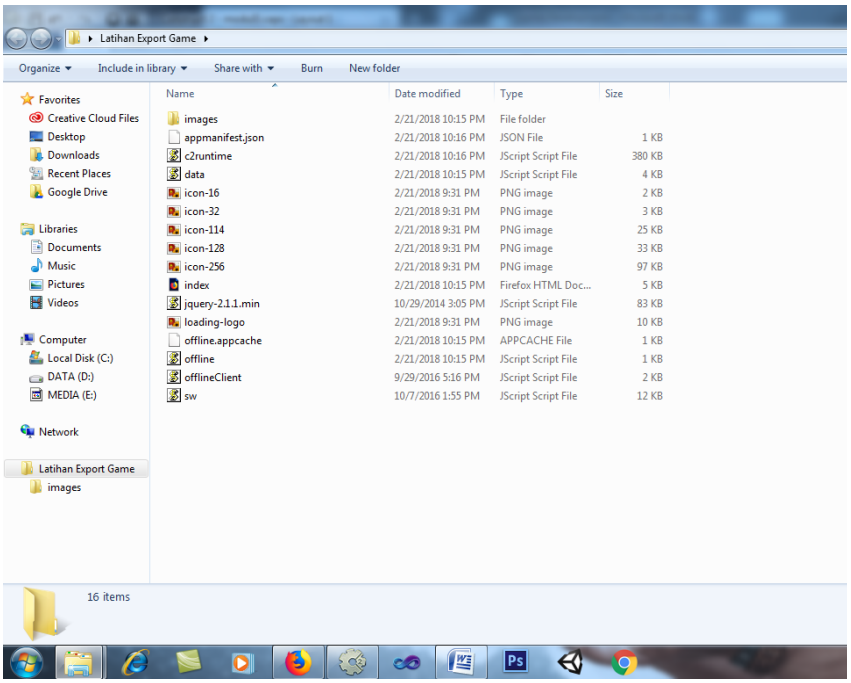
Gambar 6.5 Export game langkah ke-5

7. Setelah itu klik export dan tunggu beberapa saat sampai selesai game di compile dalam bentuk HTML 5.



Gambar 6.6 Export game langkah ke-6

8. Saat game berhasil di compile kita bisa mencoba game yang sudah di export dengan cara klik file index.html pada folder exportannya.



Gambar 6.7 Hasil Exportan HTML 5

MODUL VII

PHYSICS & PARTICLES

(Pertemuan 10)

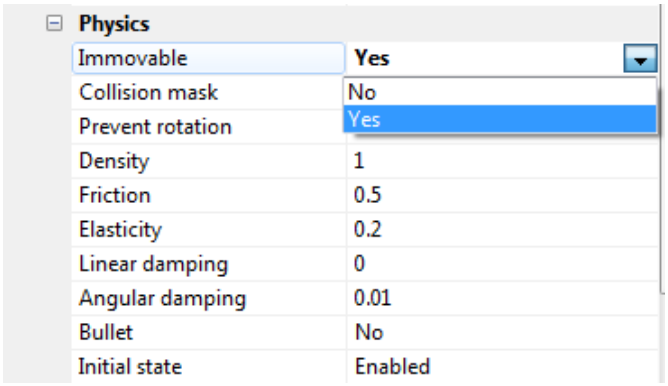
Tujuan:

1. Mahasiswa mampu mengimplementasikan prinsip fisika dengan physics dalam pembuatan game.
2. Mahasiswa mampu membuat visual effect sederhana dengan particles

7.1 Physics

Physics merupakan salah satu behavior unggulan yang dimiliki Construct 2. Jika suatu objek memiliki physics behavior, ia akan memiliki sifat-sifat fisika layaknya benda di dunia nyata. Sifat fisika tersebut misalnya gravitasi, massa, dan lain-lain. Tentu saja implementasi dari behavior ini cenderung menggunakan rumus-rumus fisika yang sudah Anda pelajari saat sekolah, sebelum memulai pratikum ada baiknya kita memahami dahulu apa fungsi-fungsi dari fitur physics Construct 2.

- **Gravitasi**
  Seperti yang kita ketehui gravitasi adalah gaya yang membuat semua benda tertarik kebawah menuju pusat bumi, begitu pula dalam Construct 2. Untuk mengutak-atik gravitasi, Anda dapat menggunakan perintah **Set Gravity** pada sebuah objek. Yang perlu diperhatikan, melakukan penggantian gravitasi akan berdampak pada semua benda di "dunia" dalam game. Untuk membuat suatu benda memiliki posisi yang tak terpengaruh oleh gravitasi dan tumbukan, maka set **Immovable** objek menjadi **Yes**. Hal itu akan membuat benda seolah-olah memiliki massa yang amat sangat besar, sehingga sulit untuk bergerak.



Gambar 7.1 Set immovable menjadi Yes

- **Collision Mask**
  Collision mask mengatur dimana saja titik-titik tabrakan sebuah objek. Secara default, collision mask akan berbentuk polygon yang memiliki lima titik. Jika diatur dalam bounding box, maka titik tabrakan akan berbentuk persegi yang menyelimuti seluruh bagian sisi objek. Sedangkan circle digunakan untuk objek yang menggelinding seperti bola.

- **Density**
  Dalam Construct 2, density digunakan untuk menunjukkan massa. Makin besar massa suatu benda, maka makin berat untuk digerakkan. Massa dapat juga dikatakan sebagai berat suatu benda, tetapi berat belum tentu dapat dibilang sebagai massa benda. Karena berat suatu benda dapat berubah tergantung gravitasi, sedangkan massa tidak.

- **Friction**
  Friction (gesekan) mempengaruhi besar pengurangan kecepatan objek karena bergesekan dengan objek lain. Hal itulan yang membuat benda yang bergerak akan berhenti pada suatu saat. Makin halus permukaan benda, maka makin kecil gesekan yang di timbulkan.

- **Elasticity**
  Suatu objek dengan elasticity tinggi akan memantul jika menabrak benda keras atau solid. Contohnya sebuah bola akan memantul kembali ke atas jika dijatuhkan ke lantai.

- **Linear Damping**
  Hukum Newton I kira-kira berbunyi, “Setiap benda akan memiliki kecepatan konstan, kecuali ada gaya dengan resultan lebih dari nol yang bekerja pada benda tersebut”. Artinya, jika resultan gaya nol, maka benda akan diam, atau bergerak dengan kecepatan konstan. Jika linear damping diaktifkan, maka gerakan benda makin lama akan makin lambat. Hingga akhirnya berhenti. Perlambatan tersebut dipengaruhi oleh beberapa hal, misalnya gravitasi, gesekan dan lain-lain.

- **Angular Damping**
  Konsep angular damping hamper sama dengan linear damping, hanya saja sifat ini hanya terjadi pada objek yang berputar. Makin tinggi angular damping, maka makin cepat benda tersebut berhenti berputar. Selain itu, angular damping tidak bergantung pada kecepatan benda.

KEGIATAN PRAKTIKUM 7.1
MEMBUAT GAME PHYSICS SEDERHANA

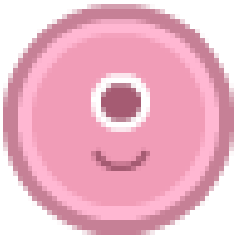
Pada praktikum 7.1 ini kita akan belajar cara membuat game physics sederhana. Dari praktikum ini di harapkan memahami seperti apa behavior physics jika di implementasikan dalam game.

1. Pertama kita buat lembar kerja baru, dengan layout size 1280, 720 px dan windows sizenya 1280, 720 px.

2. Setelah itu desainlah level game seperti dibawah ini.

Gambar 7.2 Desain Level Game Physics

Yang dimana asset dalam game tersebut antara lain



Gambar 7.3 Alien_Ball

Gambar 7.4 Papan

Gambar 7.5 Kotak

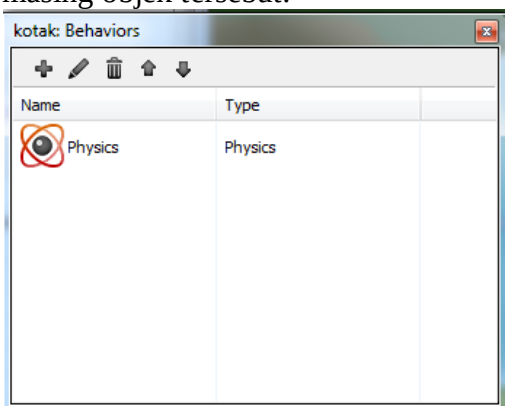
Gambar 7.6 Keranjang

Gambar 7.7 Target

Gambar 7.8 Pijakan

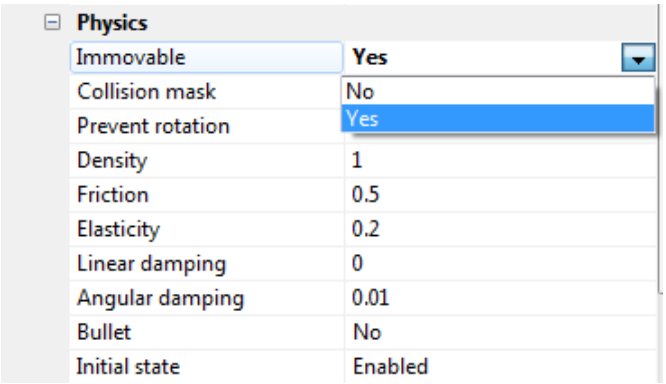
Gambar 7.9 Background

3. Tambahkan behavior physics untuk objek alien_ball,. papan, kotak, pijakan dan keranjang pada masing-masing objek tersebut.



Gambar 7.10 Tambah behavior physics

4. Cobalah playtest dan lihat hasilnya,saat dilakukan playtest terlihat objek tersebut jatuh kebawah seperti di tarik gravitasi dan menghilang keluar dari layout game. Untuk mengatasi kita bisa mengatur **Immovable** pada objek Pijakan menjadi **Yes**, seperti materi yang sebelumnya kita bahas pada materi Gravitasi yang dimana membuat benda seolah-olah memiliki massa yang amat sangat besar, sehingga sulit untuk bergerak saat Immovable di set Yes.

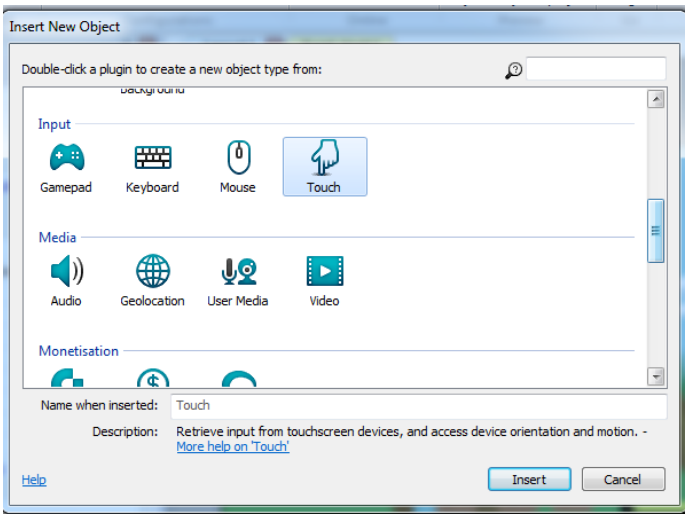


Gambar 7.11 Set immovable menjadi Yes pada objek pijakan

5. Lakukan playtest dan lihat hasilnya, disini terlihat game sudah sesuai dengan rancangan level yang kita mau, dimana physics pada objek game tersebut sudah berjalan sesuai dengan hukum fisika seperti gambar dibawah ini.

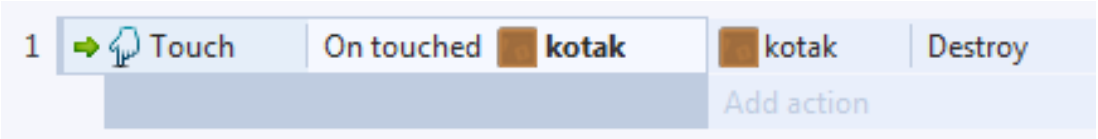
Gambar 7.12 Tampilan playtest game physics ke-1

6. Selanjutnya kita tambahkan input touch pada game yang kita buat.



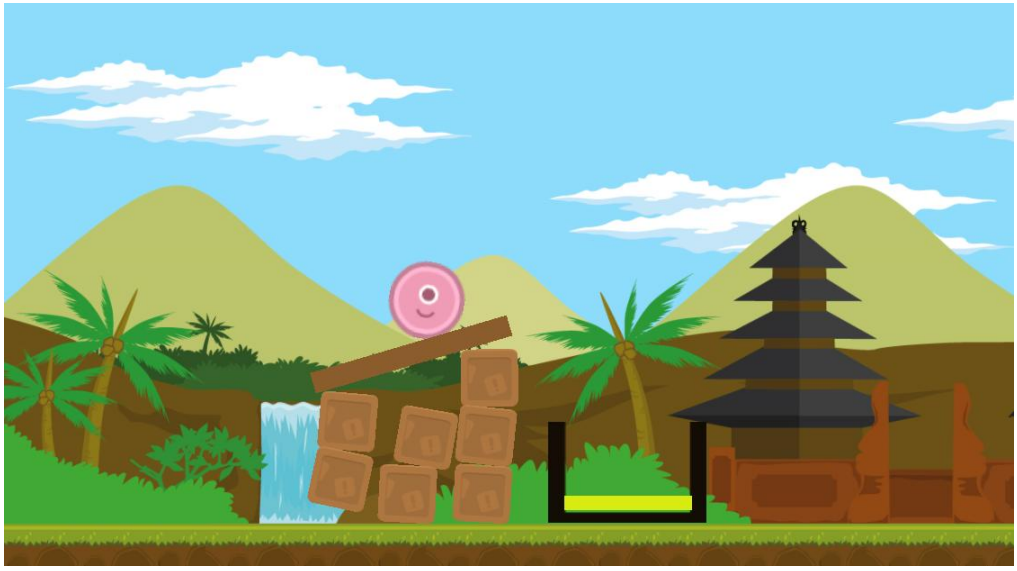
Gambar 7.13 Tambah Input Touch

7. Tambahkan kondisi pada eventsheet game yang akan kita buat, dimana kotak pada game ini jika di sentuh maka akan menghilang. Eventsheet tersebut antara lain seperti berikut



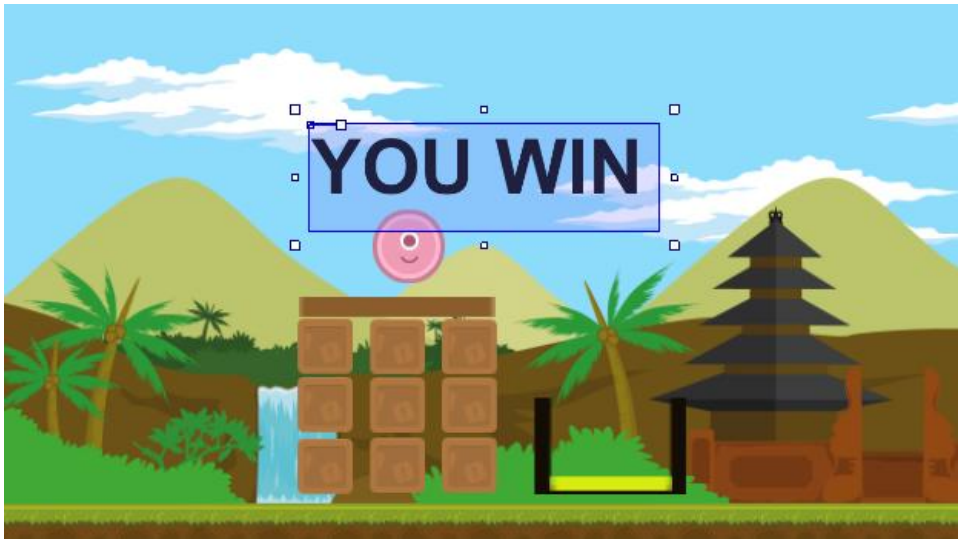
Gambar 7.14 Eventsheet touch kotak

8. Setelah itu cobalah playtest dan sentuh objek kotak pada game yang kita buat dan lihat hasilnya, disini terlihat kotak saat di sentuh kotak tersebut menghilang sehingga membuat objek yang di atasnya terlihat goyah karena kehilangan keseimbangan.



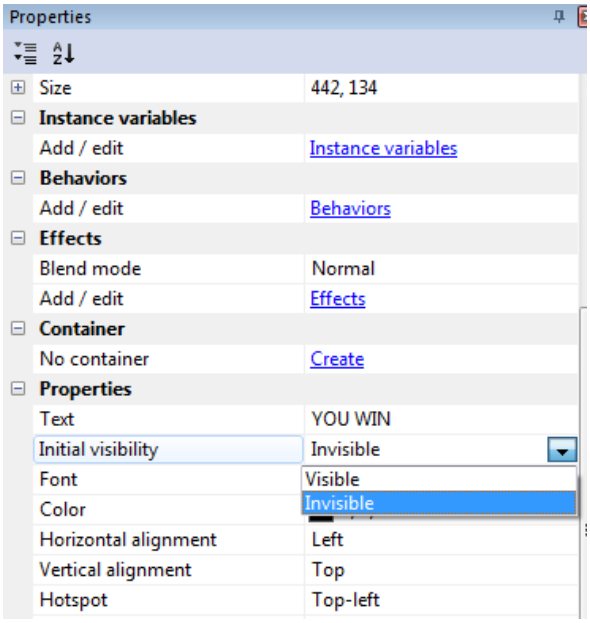
Gambar 7.15 Tampilan playtest game physics ke-2

9. Selanjutnya tambahkan objek teks dengan nama txt_win, yang dimana teks ini akan muncul saat pemain berhasil memasukan alien_ball pada keranjang. Beri tulisan "YOU WIN".



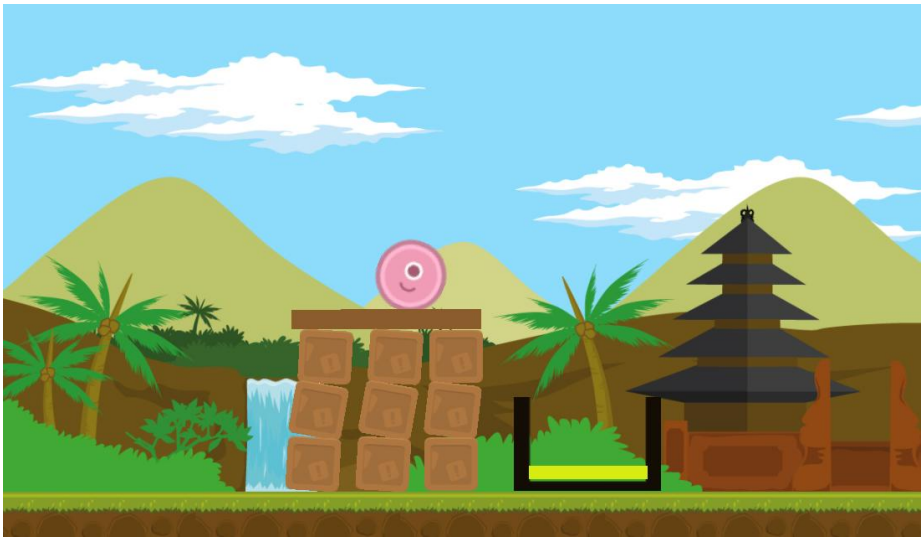
Gambar 7.16 Tambah txt_win pada game

10. Klik txt_win dan aturlah **Initial Visibility** pada properties menjadi **Invisible**, agar saat game di play txt_win tidak terlihat.



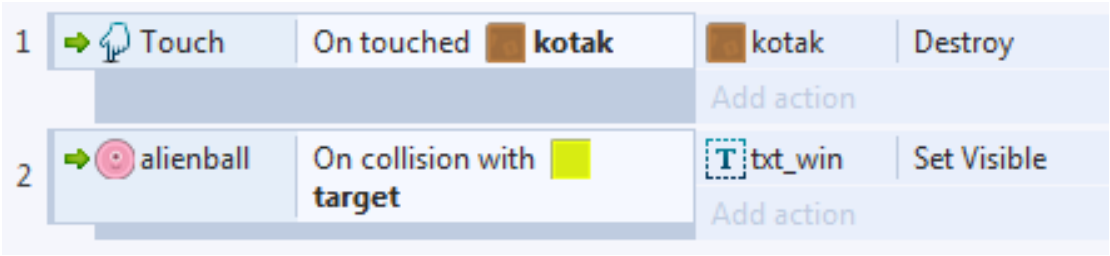
Gambar 7.17 Set Initial Visibility

11. Lakukan playtest dan lihat hasilnya, txt_win tidak terlihat saat game di play.



Gambar 7.18 Tampilan playtest game physics ke-3

12. Selanjutnya kita tambahkan kondisi ke eventsheet dimana saat pemain berhasil memasukan alienball pada keranjang dan mengenai target, maka txt_win akan muncul (terlihat) seperti dibawah ini.



Gambar 7.19 Eventsheet win game physics

13. Lakukan playtest dan cobalah memasukan alienball pada keranjang, dan terlihat txt_win muncul saat pemain berhasil memasukan alienball pada keranjang.

Gambar 7.20 Tampilan playtest win game physics

7.2 Particles

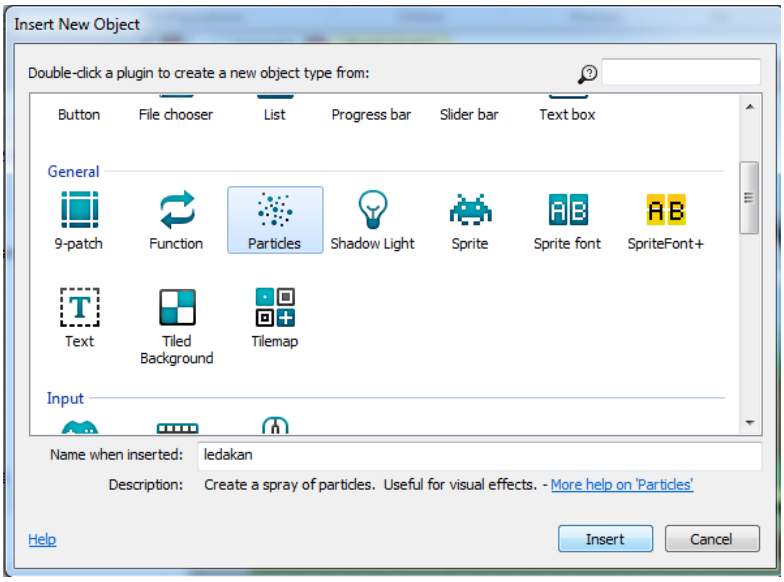
Particles merupakan objek yang ada pada Construct 2 yang dimana dapat di manfaatkan untuk membuat visual effects yang sederhana. Banyak efek yang bisa kita buat dengan objek particles antara lainya efek hujan, api, kembang api kemenangan, salju, ledakan dan lain-lain. Dengan adanya particles pada game yang kita buat, membuat terlihat lebih indah dan realistis.

KEGIATAN PRAKTIKUM 7.2
MEMBUAT VISUAL EFFECT DENGAN PARTICLES

Pada praktikum 7.2 ini kita akan belajar cara membuat visual effect sederhana dengan particles. Dari praktikum ini di harapkan mahasiswa memahami seperti apa particles jika di implementasikan dalam game.

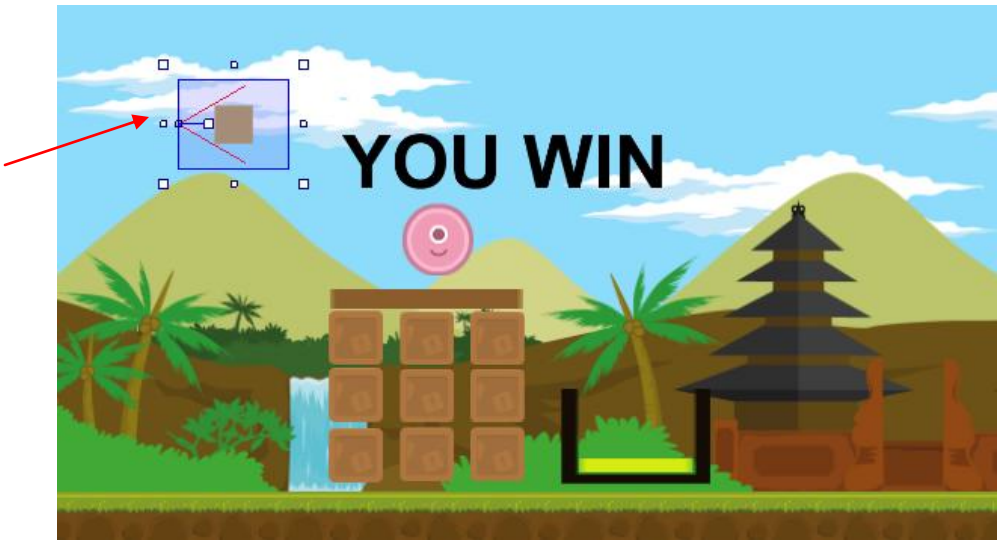
1. Pertama kita buka projek yang kita buat sebelumnya di praktikum 7.1. Disini kita akan mencoba mempercantik game tersebut particles.

2. Tambahkan objek particles pada lembar kerja kita. Berilah nama “ledakan” pada objek particles tersebut



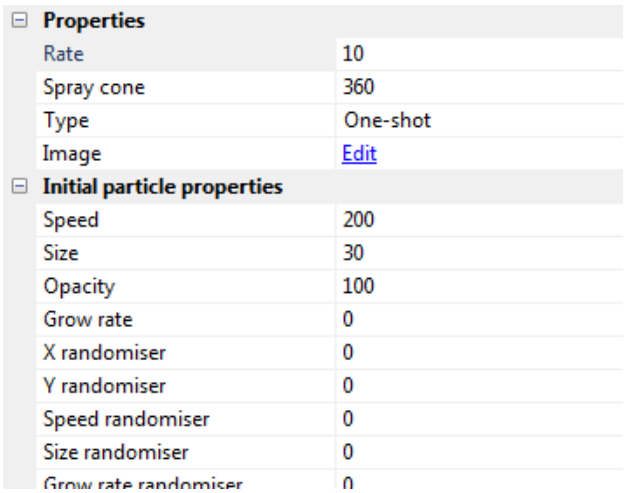
Gambar 7.21 Tambah objek particles

3. Selanjutnya tambahkan gambar untuk ledakan particles seperti gambar dibawah ini.



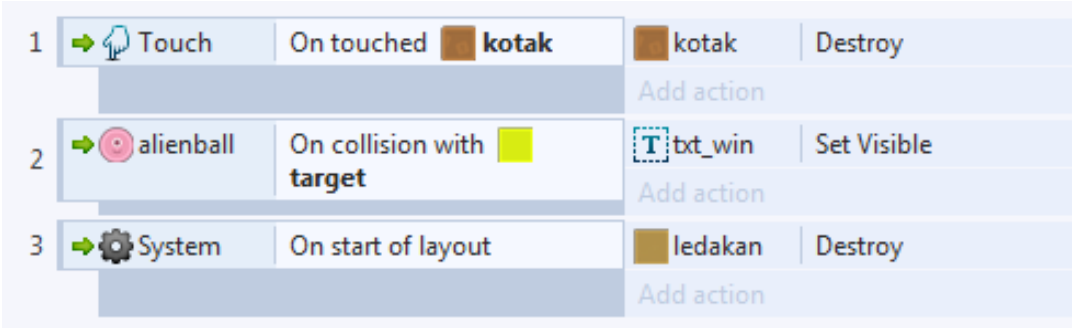
Gambar 7.22 Objek particles pada layout game

4. Klik objek ledakan dan aturlah particles pada propertiesnya seperti gambar dibawah ini.



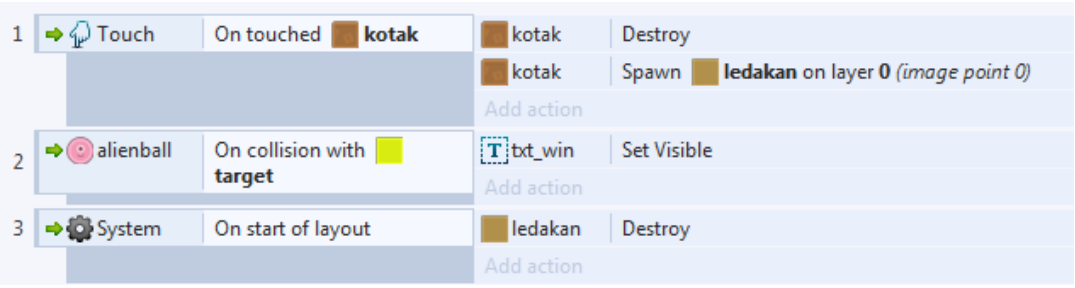
Gambar 7.23 Setting particles ledakan

5. Pada tahap ini kita buat eventsheet yang dimana saat game di restart atau dimulai pada awalnya ledakan particles kita hilangkan(dihapus) seperti kode pada baris 3 dibawah ini.



Gambar 7.24 Eventsheet start layout

6. Selanjutnya kita buat lagi eventsheet yang dimana saat kotak di touch (disentuh) maka particles ledakan muncul pada kotak yang di sentuh seperti kode baris 1 dibawah ini



Gambar 7.25 Eventsheet ledakan kotak

7. Playtest dan lihat hasilnya, di sini terlihat particles ledakan muncul saat kotak di touch(disentuh).

Gambar 7.26 Tampilan playtest particles ledakan pada kotak

8. Pada tahap ini particles ledakan sudah bisa berjalan sehingga membuat game yang kita buat lebih menarik dari sebelumnya. Untuk selanjutnya kita akan mencoba membuat ledakan kembang api kemenangan yang dimana saat tulisan win_txt muncul maka efek tersebut menghiasi tampilan game kita dengan bintang-bintang yang bertaburan. Tambahkan lagi objek particles seperti tahap sebelumnya dan berikan gambar bintang pada particles tersebut dan beri nama bintang.

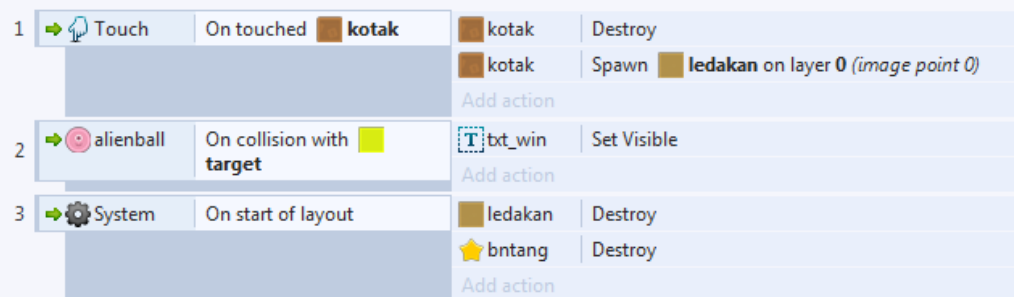
Gambar 7.27 Tambah particles bintang

9. Klik particles bintang dan aturlah particles bintang seperti berikut pada properties

Properties	
Rate	80
Spray cone	360
Type	One-shot
Image	Edit
Initial particle properties	
Speed	200
Size	32
Opacity	100
Grow rate	0
X randomiser	0
Y randomiser	0
Speed randomiser	0
Size randomiser	0
Grow rate randomiser	0

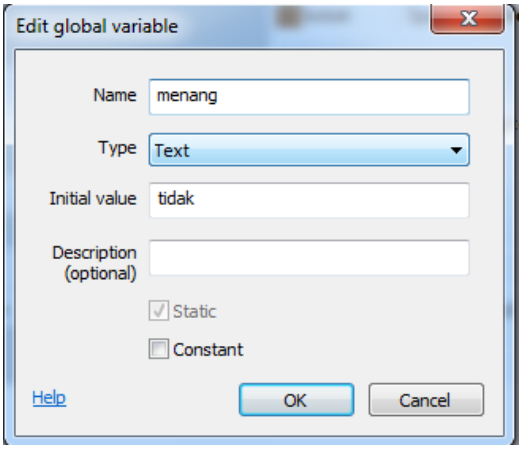
Gambar 7.28 Setting particles bintang

10. Pada tahap ini kita buat eventsheet destroy pada particles bintang yang dimana saat game di restart atau dimulai pada awalnya particles bintang kita hilangkan(dihapus) seperti kode pada baris 3 dibawah ini.



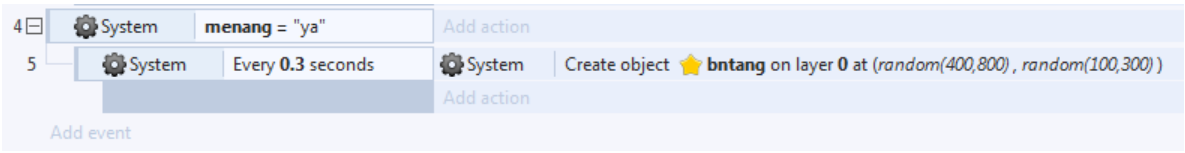
Gambar 7.29 Eventsheet destroy particles bintang

11. Selanjutnya kita buat efek kembang api saat txt_win muncul, tapi sebelumya buatlah variable menang dengan type text dan set nilainya menjadi tidak.



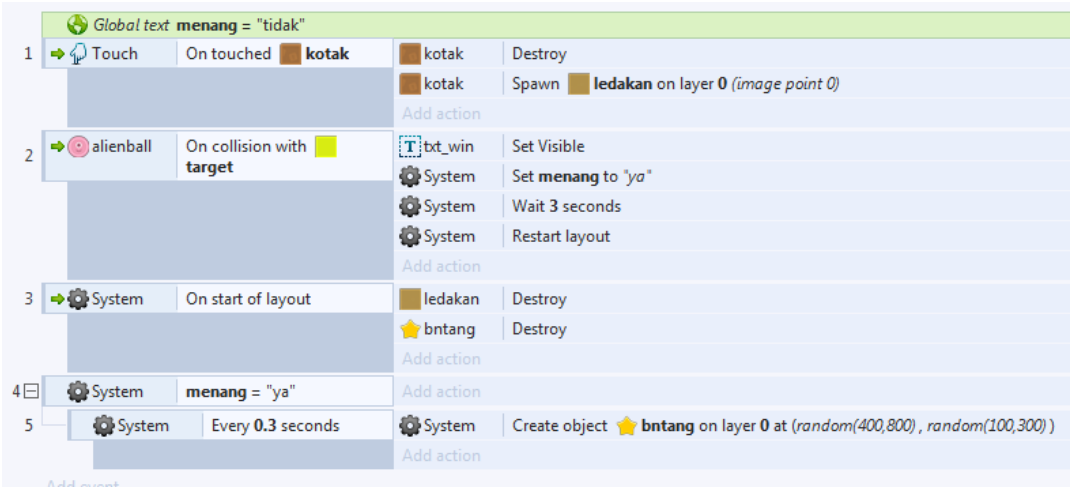
Gambar 7.30 Tambah variable menang

12. Lalu buatlah kondisi jika variable menang bernilai “ya” maka efek kembang api pada particles bintang muncul. Disini kita akan membuat if bersarang sederhana, dimana satu kondisi terpenuhi maka lakukan pengecekan if kembali seperti dibawah ini



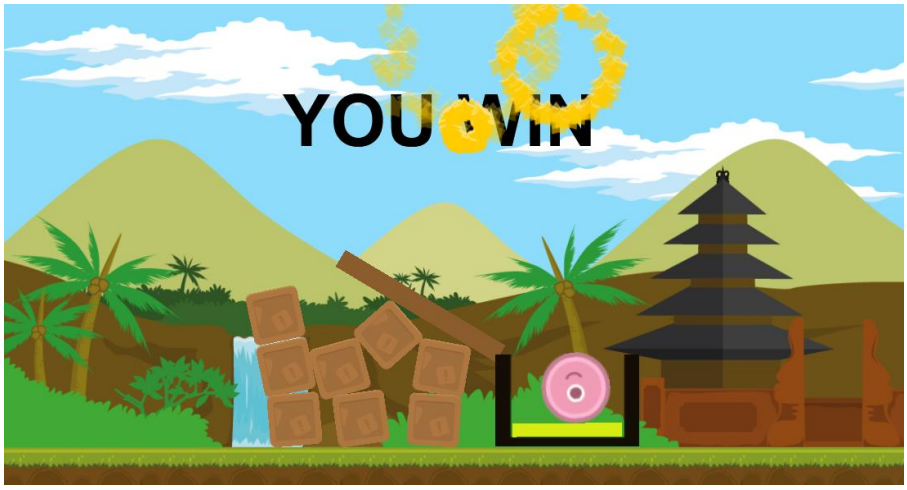
Gambar 7.31 Eventsheet if bersarang variable menang=”ya”

13. Tahap terakhir kita tambahkan kode pada eventsheet pada baris ke-2 saat kondisi pemain menang maka set variable menang bernilai “ya” dan dalam waktu 3 detik restart game kembali ke awal seperti dibawah ini.



Gambar 7.32 Eventsheet set variable menang bernilai “ya”

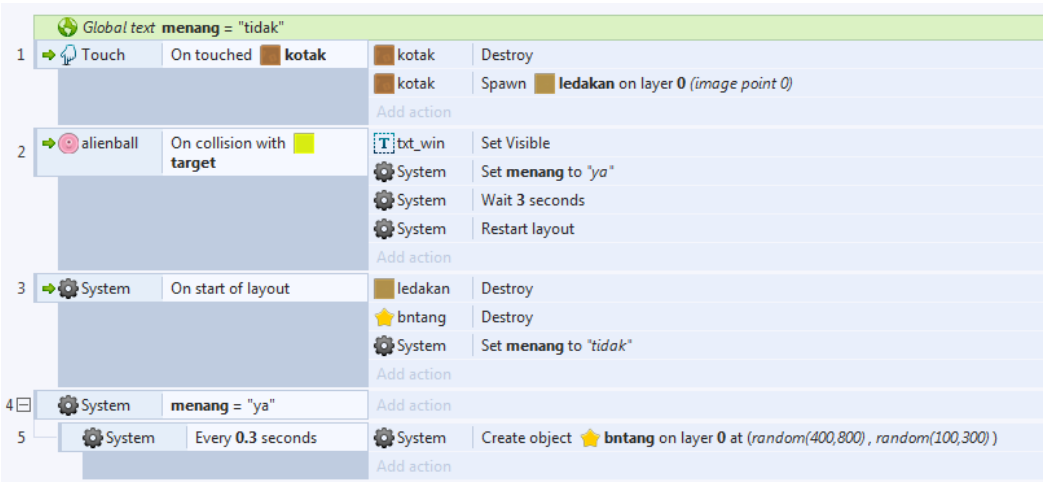
14. Cobalah playtest dan lihat hasilnya, disini terlihat saat txt_win muncul bertaburan bintang pada layar game, namun saat game restart ditemukan bug taburan bintang masih aktif.



Gambar 7.33 Tampilan playtest particles bintang saat menang

Gambar 7.34 Tampilan bug saat game di restart

15. Untuk mengatasi hal tersebut kita perlu menambahkan kode di baris 3 yaitu set variable menang menjadi “tidak”.



Gambar 7.35 Eventsheet set variable menang =”tidak”

16. Setelah itu cobalah playtest dan mainkan game tersebut, game sudah tidak di temukan bug dan berjalan dengan semestinya.

TUGAS

1. Jelaskan apa itu physics dan berikan 5 contoh game yang menggunakan physics dalam gameplaynya !
2. Cobalah set properties particles ubah typenya menjadi Continuous Spray dan analisalah dengan particles type Continuoes Spray jika di implementasikan dalam game akan lebih cocok di jadikan visual effect apa !
3. Buatlah kondisi Game Over pada project game praktikum 7.2 saat pemain jatuh ketanah !

MODUL VIII

SOUND EFFECTS

(Pertemuan 11)

Tujuan:

1. Mahasiswa memahami cara menambahkan efek suara pada game.
2. Mahasiswa memahami antara sound dan music.

DASAR TEORI

Untuk memainkan file audio didalam projek, Anda perlu memasukan plugin **Audio** terlebih dahulu, kemudian objek Audio akan memainkan file audio yang sebelumnya telah diimport ke dalam projek. Perlu diingat bahwa hanya file berekstensi **Ogg Vorbis** (.ogg) atau **MPEG-4 AAC** (.m4a) yang bisa dimainkan, karena kedua ekstensi itulah yang didukung oleh browser. Jika file Anda memiliki ekstensi lain, maka ubah dulu formatnya dengan software **audio converter.**

Didalam project bar, terdapat perbedaan fungsi antara folder **Sound** dan **Music**. Jika suatu file diletakkan dalam folder Sound, maka file tersebut harus terdownload seutuhnya dari server. Akibatnya ketika dimainkan file audio tidak akan langsung diputar (karena proses download masih berjalan), sehingga terjadi lag (keterlambatan) yang cukup menggangu, padahal game sudah dapat dimainkan. Berbeda dengan Sound, file audio didalam folder Music dapat langsung dimainkan walaupun proses download masih berjalan (streaming), sehingga audio dapat diputar saat itu juga. Namun, bisa jadi file tersebut dimainkan secara putus-putus karena koneksi yang buruk.

Untuk mengatasi keterbatasan folder Sound, dapat digunakan aksi **Preload** pada event sheet. Game tidak akan bisa dimulai dan dimainkan jika semua audio yang diberikan perintah ini belum terdownload sepenuhnya. Hal ini akan mencegah lagging ketika audio dimainkan, namun konsekuensinya, proses loading akan lebih lama.

Jika membuat game online, sebaiknya letakkan file sound effect (sfx) yang berukuran kecil pada folder Sound (dengan aksi preload), dan background music (bgm) yang berukuran lebih besar pada folder Music. Pemain cenderung batal bermain jika game terlalu lama loading.

Besar kecilnya volume audio ditentukan dengan satuan decibel (db). Secara default, volume normal audio adalah 0 db. Nilai diatas nol akan membuat suaranya lebih keras, dan nilai negative akan memelankan suaranya. Audio dengan volume -10 db akan setengah lebih pelan dari suara normal.

8.1 Audio Tags

Didalam suatu game tidak mungkin hanya mempunya satu buah music, pasti akan ada beberapa buah file audio untuk menghidupkan suasana didalam game. Permasalahannya, bisa jadi akan ada beberapa audio yang diputar dalam waktu yang bersamaan. Untuk mempermudah sistem mengenali file audio didalam project, digunakanlah istilah **tag** (label). Pemberian nama label pada audio didasarkan pada fungsi audio tersebut pada game yang dibuat. Misalnya, music yang dimainkan saat karakter meloncat, maka nama pelabelannya menjadi **"Jump**". Jika terdapat beberapa file audio yang memiliki fungsi yang sama, maka dapat diberi label yang sama pula.

Penggunaan label dalam game cukup penting, karena semua kondisi dan aksi akan menggunakan label untuk memilih file audio mana yang diinginkan. Misalnya jika ingin mengganti background music (bgm), maka bgm yang lama perlu dimatikan terlebih dahulu dengan perintah **Stop ("label_audio").** Sebelum kita pergi ke materi lebih lanjut, ada baiknya kita mengetahui apa saja fungsi setiap perintah pada Audio.

1. **Conditions**
   - **Is any playing:** bernilai benar jika ada audio yang sedang dimainkan.
   - **Is silent:** bernilai benar jika salah satu onjek di set silent dengan aksi silent.
   - **Is tag playing:** bernilai benar jika audio dengan label tertentu sedang dimainkan.
   - **On ended:** bernilai benar jika audio dengan label tertentu selesai dimainkan (tak berpengaruh untuk loop playing).
   - **Preloads complete:** bernilai benar jika semua audio telah selesai di download.

2. **Actions**
   - **Play/Play (by name):** memainkan file audio. Pada saat inilah label diberikan.
   - **Preload/Preload (by name):** mendownload lagu, sehingga tidak terjadi lagging saat game dimainkan.
   - **Set looping:** memutar file audio berulang-ulang tanpa berhenti.
   - **Set master volume:** set volume semua file audio.
   - **Set muted:** mematikan suara, namun masih tetap dimainkan.
   - **Set paused:** menghentikan sementara.
   - **Set silent:** mematikan suara semua audio.
   - **Set volume:** mengatur volume secara spesifik.
   - **Stop:** menghentikan audio secara spesifik.
   - **Stop all:** menghentikan semua audio.

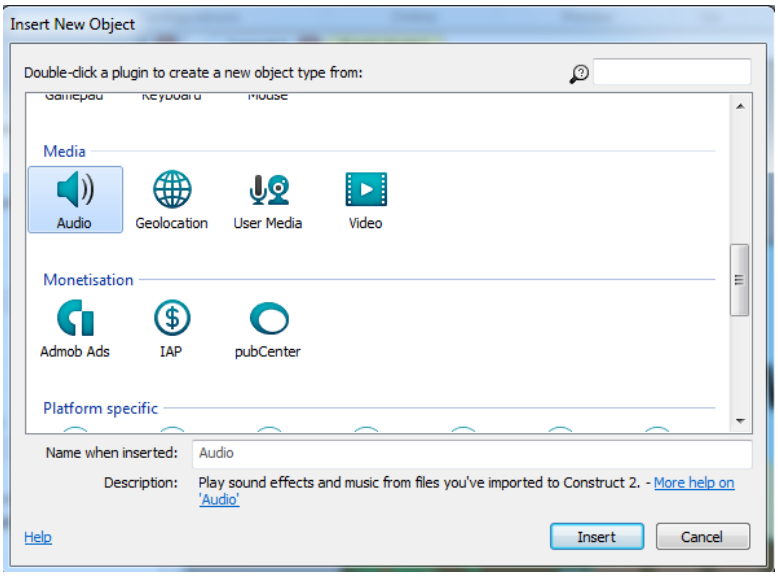
3. **Expressions**
   - **Duration (Tag):** mencari durasi file audio dengan label tertentu.
   - **Master Volume:** mencari volume master suara.
   - **Volume (Tag):** mencari volume file audio dengan label tertentu.

KEGIATAN PRAKTIKUM 8.1
MENAMBAHKAN AUDIO PADA GAME

Pada praktikum 8.1 ini kita akan belajar cara menambahkan audio pada game yang kita buat. Dari praktikum ini di harapkan mahasiswa memahami cara menambah audio pada game.

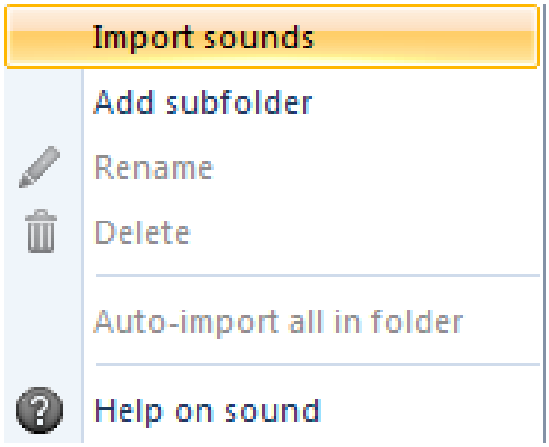
1. Pertama kita buka projek yang kita buat sebelumnya di praktikum 7.2. Disini kita akan mencoba menambah audio pada game tersebut.

2. Tambahkan objek Audio pada lembar kerja kita.



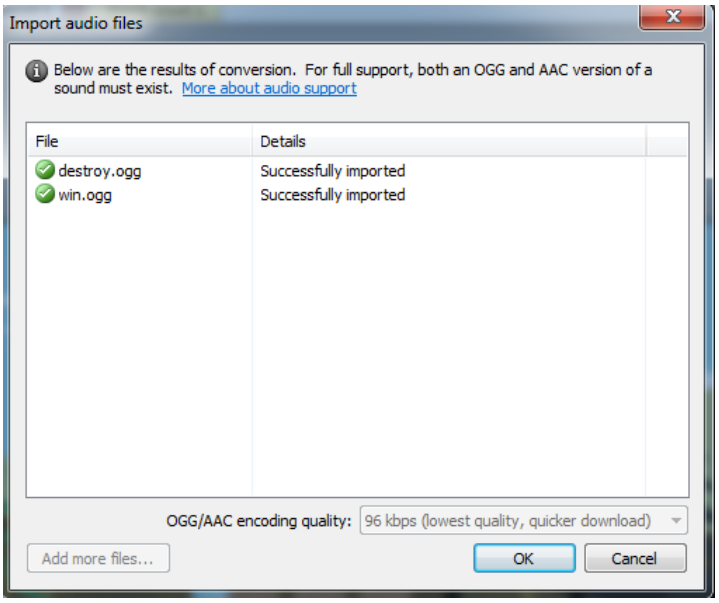
Gambar 8.2 Tambah object Audio

3. Import file **win**, dan **destroy** dengan cara klik kanan pada folder **Sounds** pada Project Bar, kemudian file **bgmusic** pada folder **Music**.



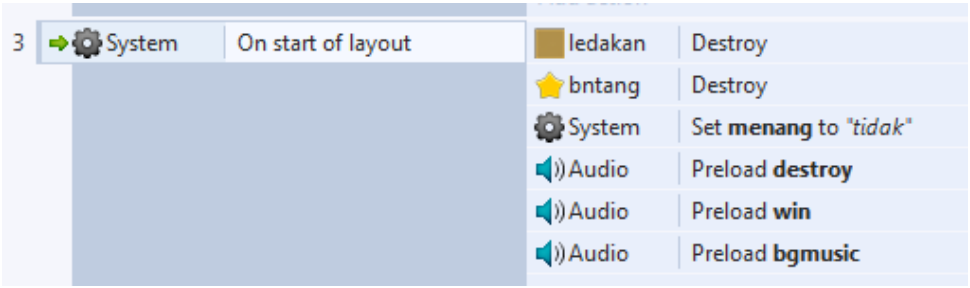
Gambar 8.3 Import file dengan klik kanan pada folder Sounds

4. Kemudian akan muncul kotak dialog import audio files. Set kualitas suara pada **OGG/ACC encoding quality**, makin banyak angkanya maka makin bagus kualitas suara yang dihasilkan. File yang berhasil dimport akan bertuliskan **"Successfully imported"** pada bagian **Details.**



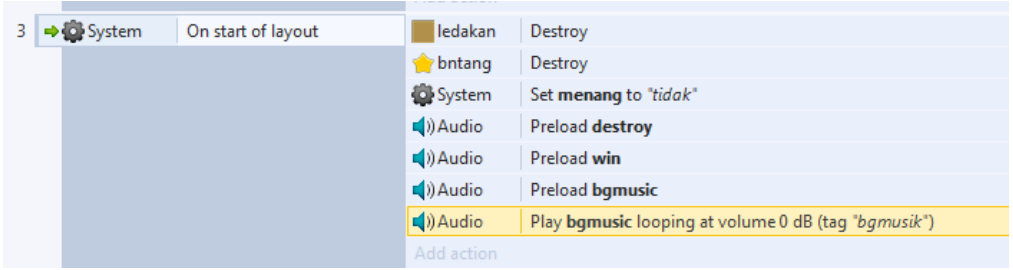
Gambar 8.4 Kotak dialog import audio files

5. Selanjutkan masukan kode **Preload** semua file audio untuk mencegah lagging pada event sheet pada saat game di restart atau dimulai seperti dibawah ini.



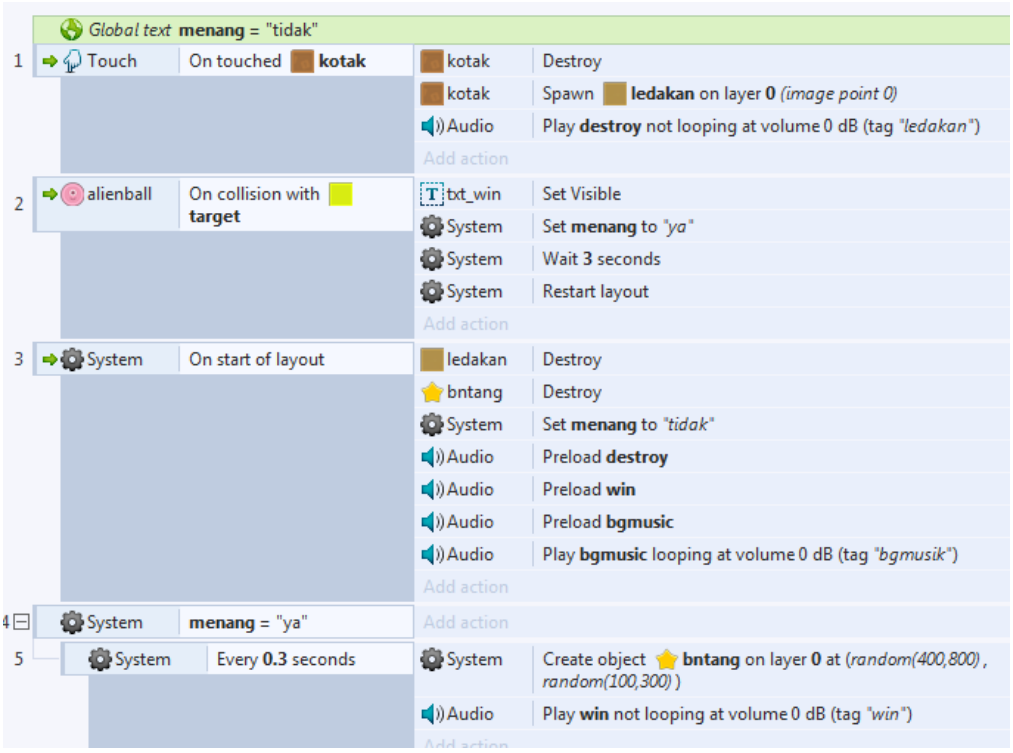
Gambar 8.5 Event sheet Preload Audio

6. Untuk memainkan audio **bgmusic** tambahkan kode **Play** pada event sheet seperti dibawah ini. Set **Loop = Looping** dan berikan **Tag = "bgmusic".**



Gambar 8.6 Event sheet Play Audio "bgmusic"

7. Selanjutnya lakukan hal yang sama seperti diatas pada Audio destroy dan win, tempatkan pada posisi yang sesuai. Untuk Audio destroy dan win set **Loop = Not Looping** dan berikan Tag yang sesuai.



Gambar 8.7 Event sheet Play Audio

TUGAS

1. Buatlah tombol play sound dan stop sound pada game yang kalian buat !

MODUL IX

SAVE & LOAD

(Pertemuan 12)

Tujuan:

1. Mahasiswa memahami cara membuat save & load dalam Construct 2
2. Mahasiswa memahami antara Save State & Local Storage

DASAR TEORI

Sebuah game pastilah memiliki save dan load, karena tidak mungkin pemain akan terus menerus memainkan game hingga selesai/tamat. Biasanya saat kita bermain game yang berisi level, pemain hanya hanya bisa memilih level yang sudah terbuka. Sebuah level akan terkunci jika pemain belum pernah mencapai level tersebut. Fitur ini dibuat agar pemain tidak harus selalu mengulang dari level pertama jika ingin mencapai level terakhir. Untuk membuat fitur tersebut kita harus bisa menyimpan informasi level tertinggi yang sudah dibuka oleh pemain, sehingga jika pemain menutup permainan dan membuka permainan di lain hari, data informasi level tertinggi tersebut tetap ada. Didalam Construct 2 ada dua tipe penyimpanan data yaitu:

- **Save State**
  Contruct akan menyimpan kondisi terakhir dari setiap objek yang ada dalam permainan, mulai dari posisi, frame animasi, nilai variable, dan lain-lain. Saat kita panggil fitur **Load State**, kondisi permainan akan langsung dikembalikan ke kondisi terakhir saat fitur save state ini di panggil.

- **Local Storage**
  Dalam Construct kita di sediakan objek Local Storage yang dimana bisa menyimpan kombinasi **"key"** dan **"value"** tertentu sesuai dengan kebutuhan kita. Keuntungan utama Local Storage adalah karena hanya data yang kita simpan yang akan di load, proses loading akan jauh lebih cepat dibandingkan fitur Save State yang menyimpan informasi semua objek dalam permainan. Fitur Local Storage ini bisa digunakan mulai Construct 2 versi 206 atau lebih, sebelumnya fitur ini bernama Web Storage.

Kedua jenis penyimpanan data tersebut memiliki kelebihan dan kekurangan masing-masing. Berikut adalah beberapa contoh kasus dan jenis penyimpanan apa yang cocok untuk digunakan. Untuk permainan action atau petualangan yang memiliki banyak musuh atau objek acak lainnya dan proses load akan membawa kita langsung ke tengah permainan, fitur Save State akan memudahkan kita karena kita tidak perlu menyimpan satu per satu informasi objek seperti posisi, nyawa, dan lain-lain.

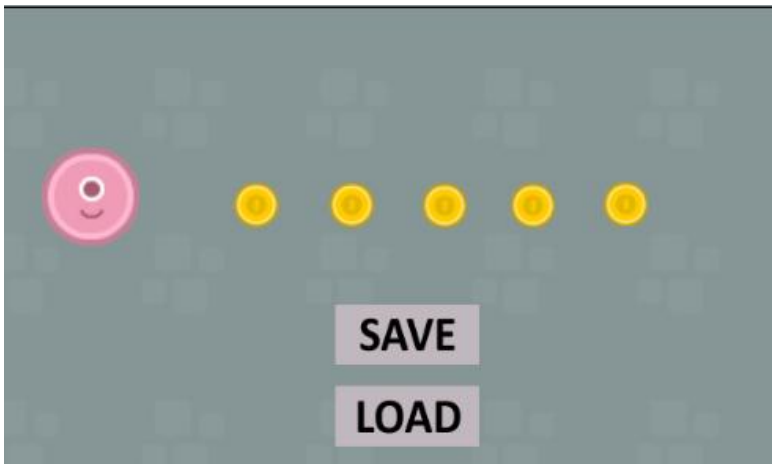
Untuk permainan dengan sesi permainan singkat dan proses load tetap membuat kita memulai dari awal level seperti permainan puzzle, quiz, atau balapan mobil, kita cukup menggunakan fitur Local Storage.

KEGIATAN PRAKTIKUM 9.1
MENYIMPAN PERMAINAN DENGAN SAVE STATE

Pada praktikum 9.1 ini kita akan belajar cara menyimpan permainan dengan Save State. Dari praktikum ini di harapkan mahasiswa memahami cara menggunakan Save State.

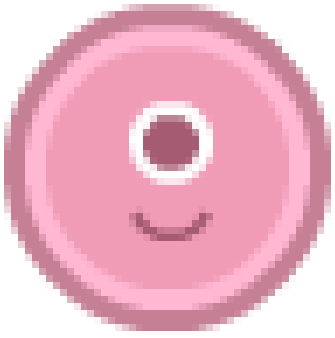
1. Pertama kita buat project baru dengan Layout size 1280 x 720 px dan Windows size 1280 x 720 px.

2. Setelah itu buatlah desain level seperti berikut

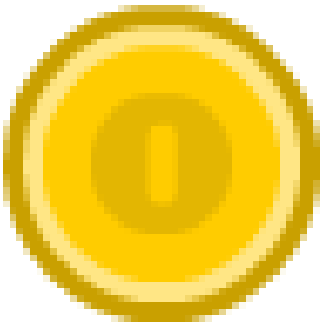


Gambar 9.1 Desain Level Game Save State

Yang dimana asset dalam game tersebut antara lain



Gambar 9.2 alien_ball2



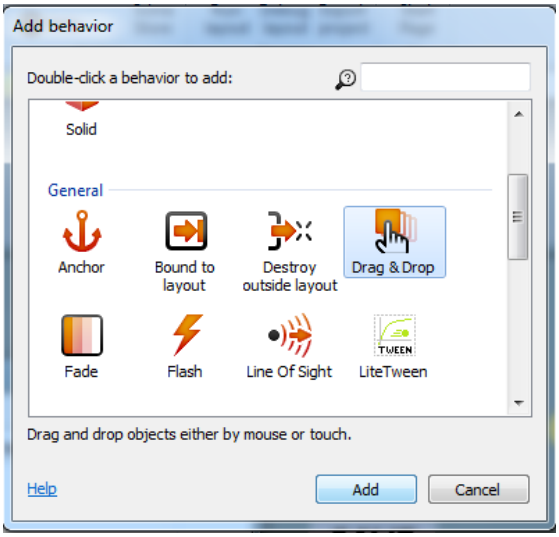
Gambar 9.3 coin

Gambar 9.4 bg_castle

Gambar 9.5 tb_save

Gambar 9.6 tb_load

3. Tambahkan behavior Drag & Drop pada alien_ball2 agar pemain bisa diseret dan di lepas layaknya bermain puzzle.

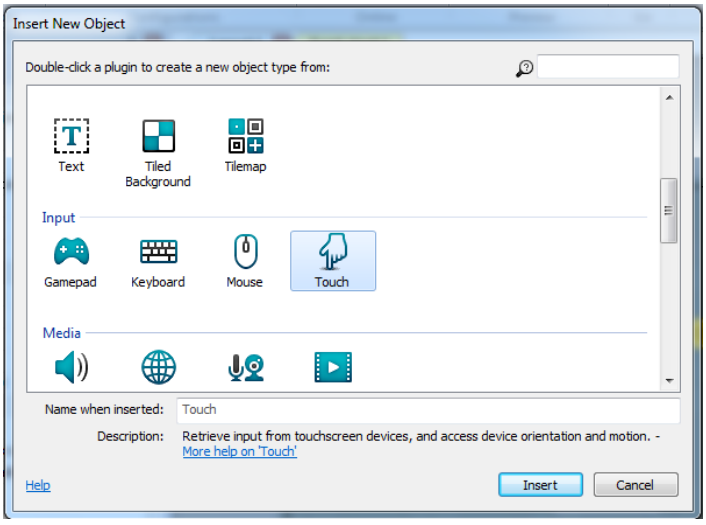


Gambar 9.7 Tambah behavior Drag & Drop pada alien_ball2

4. Buatlah kondisi dimana saat pemain menabrak coin, coin tersebut menghilang(destroy).

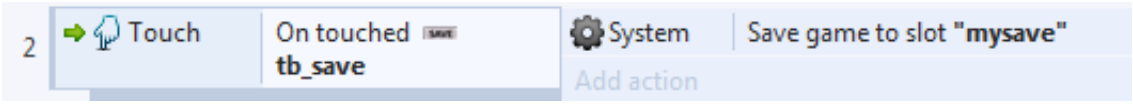
Gambar 9.8 Event Sheet Destroy Coin

5. Selanjutnya tambahkan object touch pada lembar kerja.



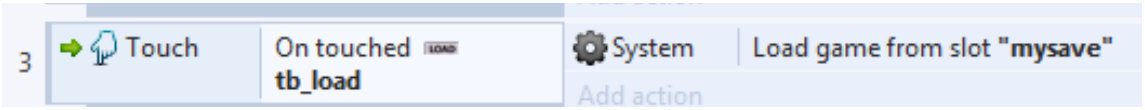
Gambar 9.9 Tambah Object Touch

6. Pada tahap ini kita akan membuat kondisi agar permainan bisa di save saat tb_save di touch (sentuh). Buatlah kode seperti dibawah ini pada event sheet.



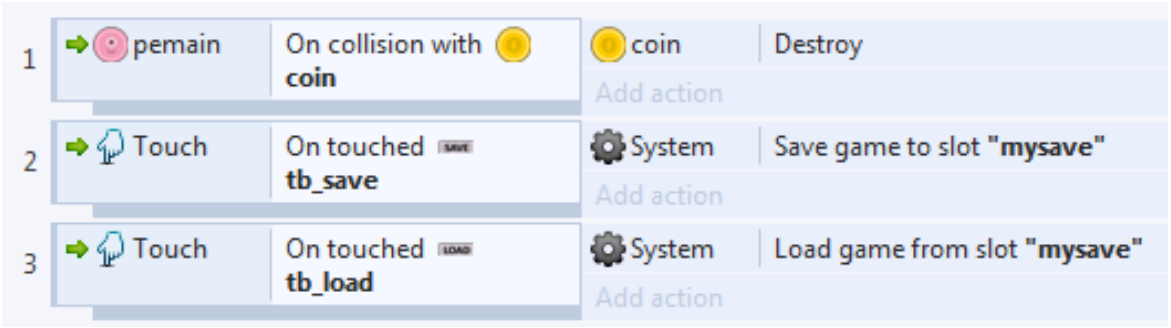
Gambar 9.10 Event Sheet Save Game

7. Setelah membuat event sheet untuk save game, selanjutnya kita tambahkan kondisi saat pemain touch (sentuh) tb_load , maka load game yang sudah di save sebelumnya. Masukan kode untuk load game pada event sheet seperti dibawah ini.



Gambar 9.11 Event Sheet Load Game

Untuk load game harap diperhatikan nama slot yang di load harus sesuai dengan nama slot save game sebelumnya yang dimana nama slot save game sebelumnya **"mysave",** karena hal ini akan mempengaruhi jalan tidaknya event sheet yang di buat.



Gambar 9.12 Event Sheet Seluruhnya Praktikum 9.1

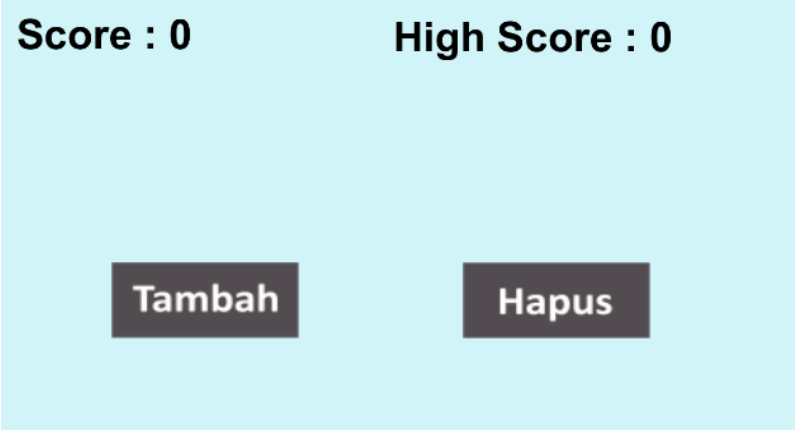
8. Cobalah playtest lalu gerakan pemain. Setelah itu tekan tb_save lalu mainkan tabrak coin yang ada sampai habis dan coba tekan tb_load untuk load game saat di save.

KEGIATAN PRAKTIKUM 9.2
MEMBUAT HIGHSCORE DENGAN LOCAL STORAGE

Pada praktikum 9.2 ini kita akan belajar cara membuat highscore dengan Local Storage. Dari praktikum ini di harapkan mahasiswa memahami cara menggunakan Local Storage dalam menyimpan highscore.

1. Pertama kita buat project baru dengan Layout size 1280 x 720 px dan Windows size 1280 x 720 px

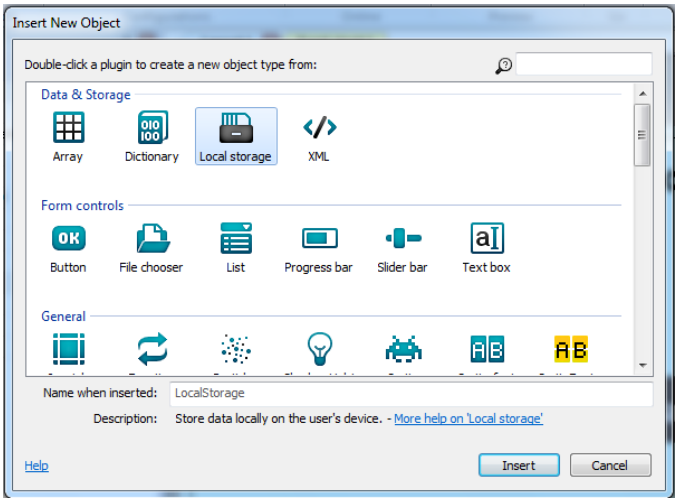
2. Setelah itu buatlah desain level game seperti dibawah ini.



Gambar 9.13 Desain Level Game Local Storage

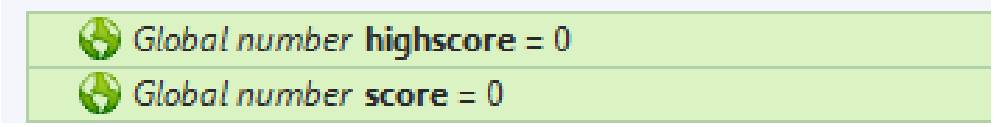
Yang dimana dalam game di atas terdiri dari tb_tambah, tb_hapus, txt_Score, txt_Highscore, dan background.

3. Tambahkan object Touch dan Local Storage pada lembar kerja



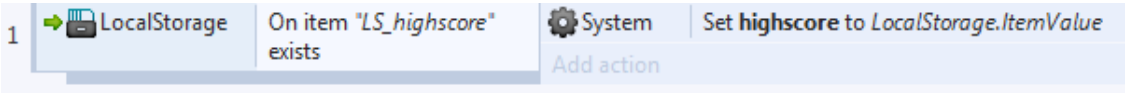
Gambar 9.14 Tambah Object Local Storage

4. Selanjutnya buatlah variable score dan highscore. Set nilainya menjadi 0 dengan type number.



Gambar 9.15 Tambah variabel score dan highscore pada event sheet

5. Pada event sheet tambahkan kode untuk menyimpan nilai variable highscore ke local storage seperti dibawah ini dan berikan nama Local Storagenya "LS_highscore".

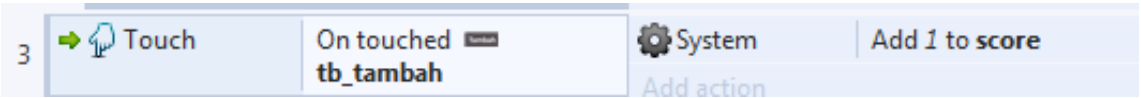


Gambar 9.16 Event Sheet menyimpan nilai variable highscore ke local storage

6. Selanjutnya tambahkan kode dengan kondisi saat game di restart/mulai cek nilai yang ada dalam Local Storage pada event sheet seperti dibawah ini.

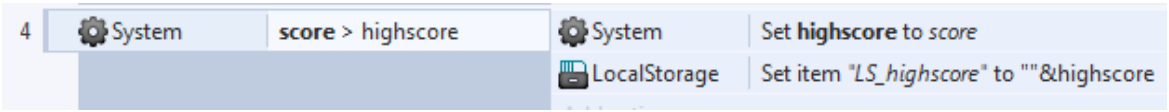
Gambar 9.17 Event Sheet mengecek nilai local storage

7. Pada tahap ini tambahkan kondisi saat tb_tambah di Touch (sentuh), maka nilai variable score bertambah dengan nilai 1.



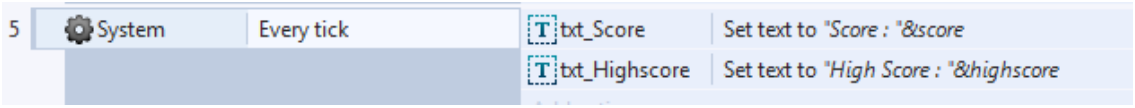
Gambar 9.18 Event Sheet tambah score

8. Tambahkan lagi kode dengan kondisi jika nilai score > highscore maka, set nilai variable highscore dengan nilai variable score dan set nilai local storage "LS_highscore" dengan nilai variable highscore.



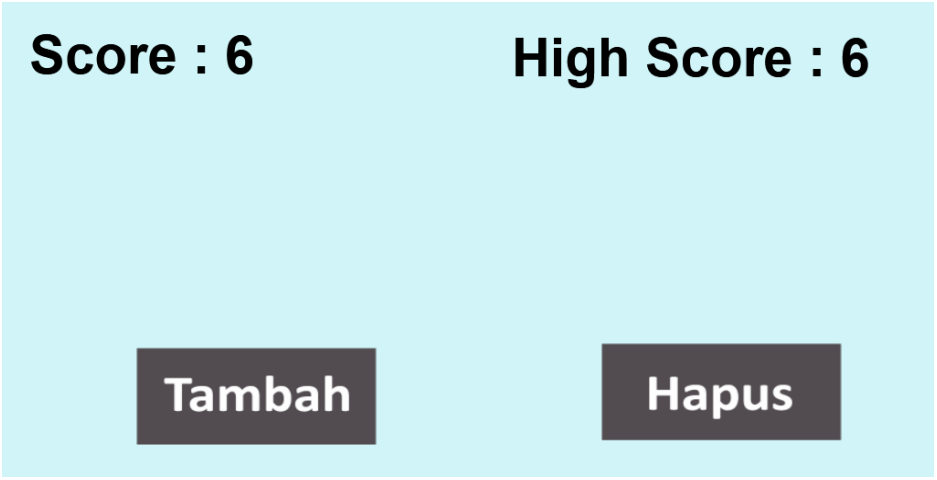
Gambar 9.19 Event Sheet set nilai "LS_highscore" dengan nilai variable highscore

9. Selanjutnya tambahkan kode untuk membuat agar txt_Score dan txt_Highscore bisa realtime mengupdate nilai yang di simpan pada event sheet seperti dibawah ini.

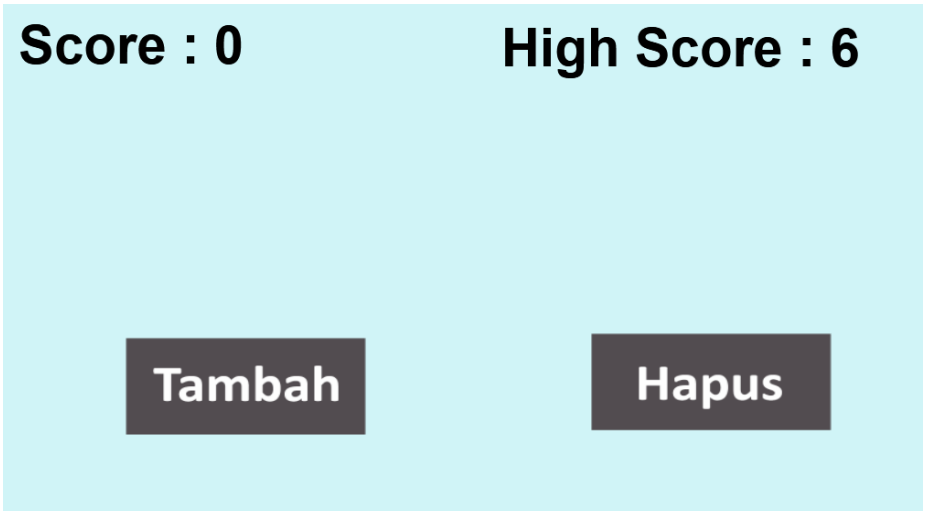


Gambar 9.20 Event Sheet set nilai txt_Score dan txt_Highscore secara realtime

10. Pada tahap ini game sudah bisa di playtest untuk mengetahui seperti apa hasil dari event sheet yang sudah di buat.

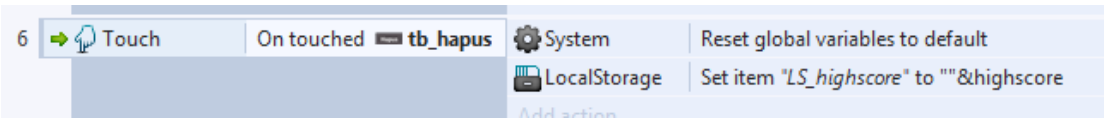


Gambar 9.21 Tampilan Playtest Praktikum 9.2 saat menambah score

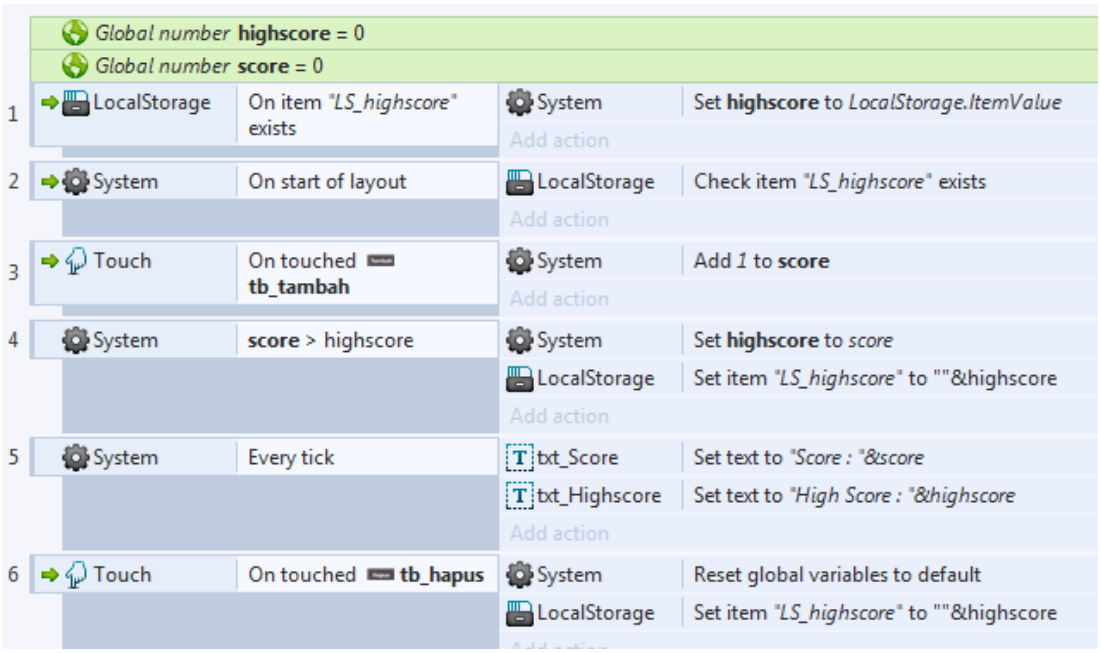


Gambar 9.22 Tampilan Playtest Praktikum 9.2 saat restart game

11. Dari playtest yang sudah di coba game sudah bisa berjalan sesuai dengan event sheet yang dibuat. Pada langkah terakhir ini kita akan menambahkan kode dengan kondisi saat tb_hapus di Touch (sentuh), maka reset nilai yang tersimpan menjadi 0.

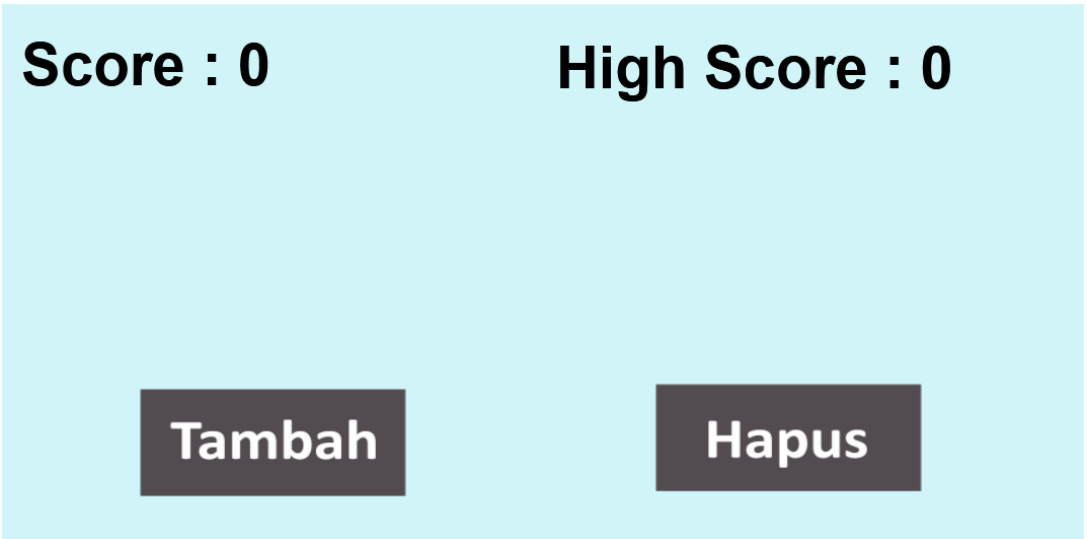


Gambar 9.23 Event Sheet reset nilai



Gambar 9.24 Event Sheet seluruhnya praktikum 9.2

12. Selanjutnya playtest dan lihat hasilnya saat tb_hapus di Touch (sentuh).



Gambar 9.25 Tampilan playtest saat tb_hapus di Touch (sentuh)

TUGAS

1. Jelaskan apa perbedaan Save State dan Local Storage !
2. Berikan contoh game yang menggunakan Save Stage dan Local Storage !
3. Buatlah menu select level seperti berikut gunakan Local Storage untuk membuatnya !

MODUL X

EXPORT GAME MOBILE

(Pertemuan 12)

Tujuan:

1. Mahasiswa memahami cara export game ke mobile

DASAR TEORI

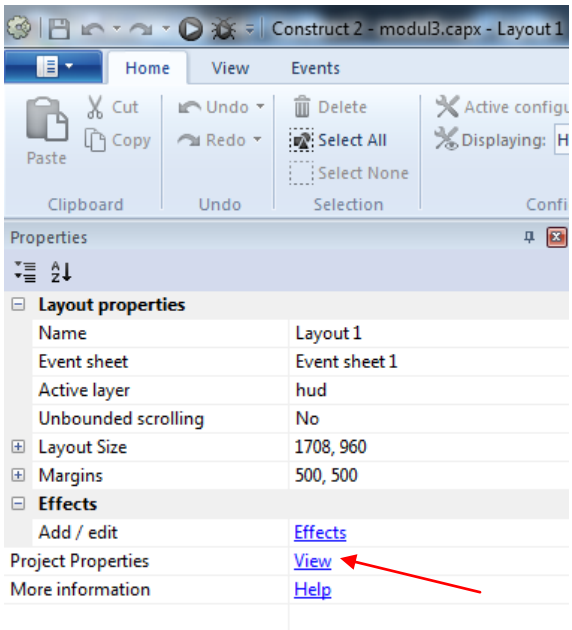
Perkembangan teknologi kini berada pada babak baru dengan kehadiran smartphone yang mempermudah kehidupan sehari-hari masyarakat. Smartphone ini mudah dibawa kemanapun dan banyak dikembangkan berbagai macam aplikasi dan game. Banyak aplikasi dan game yang beredar baik di playstore maupun appstore karena melihat begitu besar peluang pasar untuk menjual aplikasi dan gamenya. Construct 2 adalah salah satu game engine yang menjadi pilihan game developer yang mampu membantu programmer pemula maupun expert dapat dengan mudah membuat sebuah game mobile baik itu android, ios, windows, dan tizen. Dalam mengeksport game dari Construct ke mobile ada beberapa cara baik secara online maupun offline. Pada Modul X ini kita akan membahas hanya dua cara dalam mengeksport game mobile yaitu dengan cara offline melalui handphone mengcompile game yang kita buat menggunakan aplikasi C2 Buildozer dan cara online melalui web Cocoon Io.

KEGIATAN PRAKTIKUM 10.1
EXPORT GAME DENGAN C2 BUILDOZER

Pada praktikum 10.1 ini kita akan belajar cara mengexport game ke mobile secara offline melalui handphone dengan aplikasi C2 Buildozer.

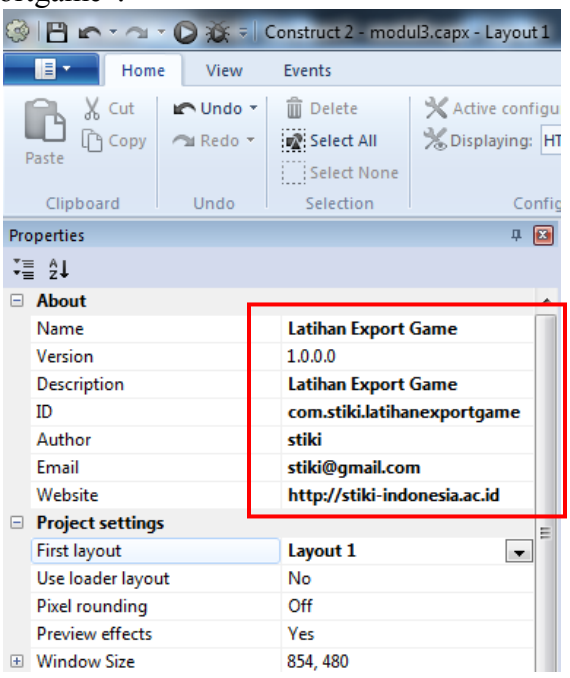
1. Pertama bukalah project game yang akan di export.

2. Klik view pada properties bar



Gambar 10.1 Export game dengan C2 Buildozer langkah ke-1

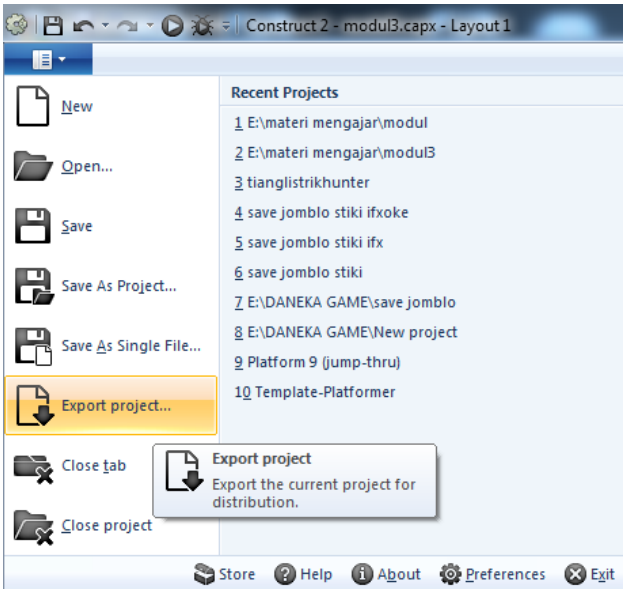
3. Isilah data about pada properties dengan lengkap seperti dibawah ini jangan sampai ada yang kosong. Pada bagian ID di harapkan defaultnya wajib diubah contoh "com.stiki.latihanexportgame".



Gambar 10.2 Export game dengan C2 Buildozer langkah ke-2

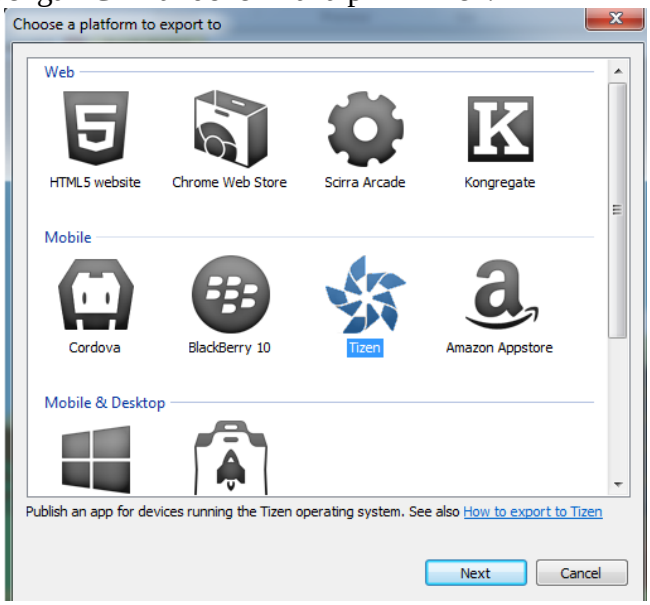
Pada bagian First Layout diharapkan di set sesuai layout apa yang akan kita play saat game mulai contoh kita set "Layout 1" sebagai layout awal yang kita play.

4. Setelah itu klik File > Export project



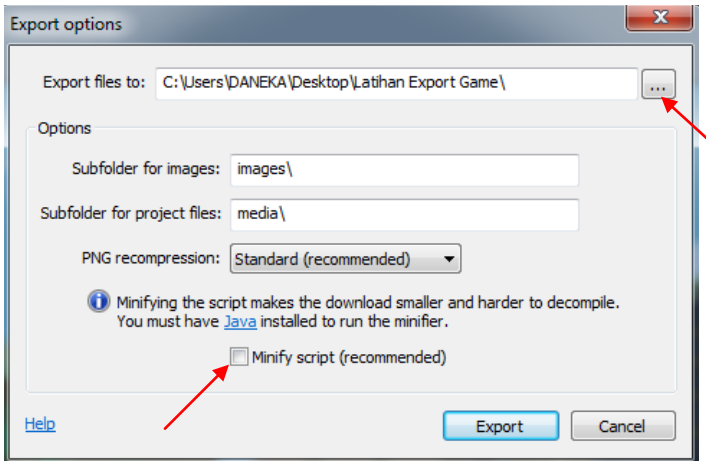
Gambar 10.3 Export game dengan C2 Buildozer langkah ke-3

5. Pada tahap ini kita akan melihat banyak pilihan untuk export game, karena kita ingin export game kita dengan C2 Buildozer maka pilih Tizen.



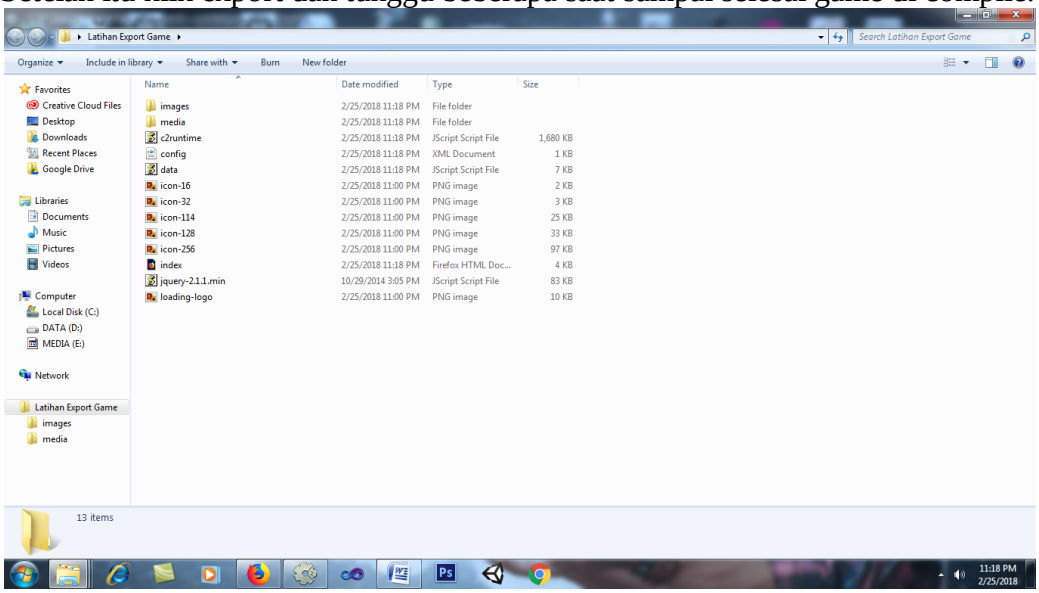
Gambar 10.4 Export game dengan C2 Buildozer langkah ke-4

6. Dalam Export files to aturlah dimana game kita akan diexport dan bagian manify script tidak perlu di centang.



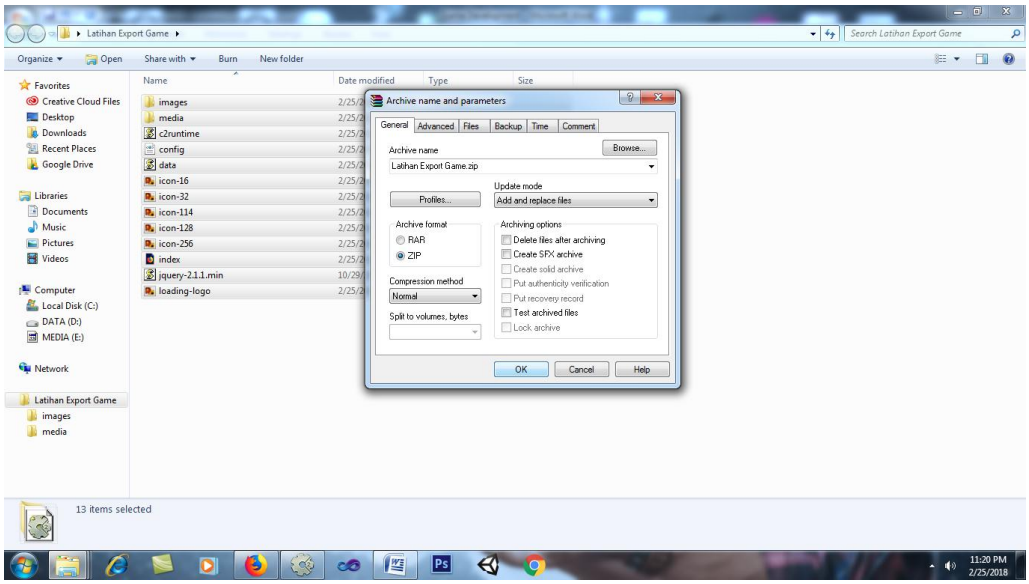
Gambar 10.5 Export game dengan C2 Buildozer langkah ke-5

7. Setelah itu klik export dan tunggu beberapa saat sampai selesai game di compile.

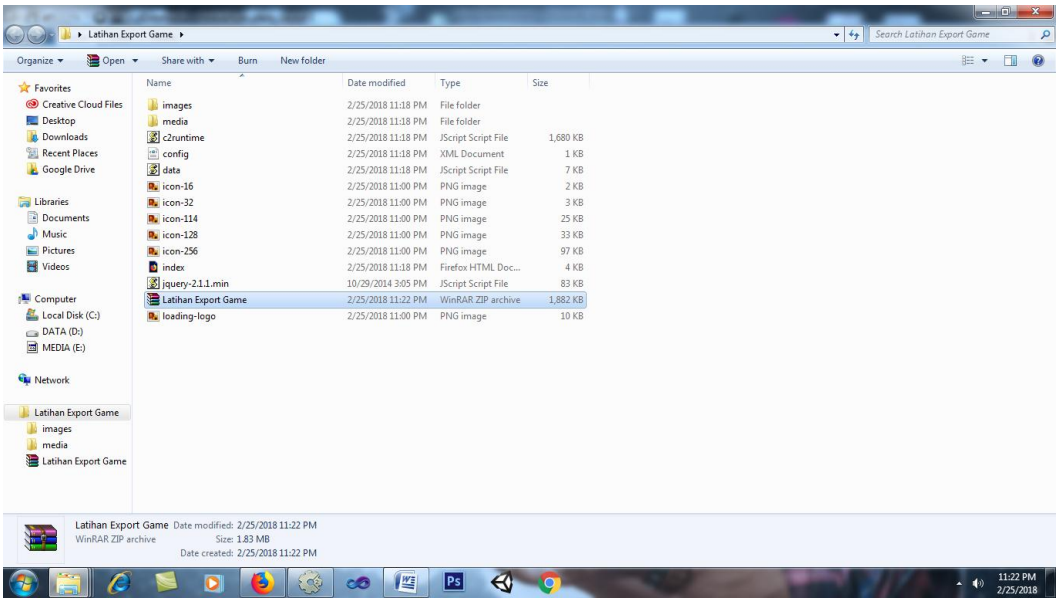


Gambar 10.6 Hasil Export

8. Saat game berhasil di compile selanjutnya select semua hasil exportnya dan jadikan dalam satu folder zip.

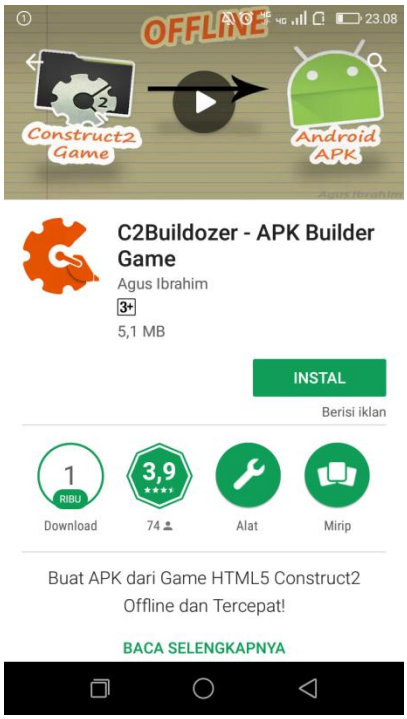


Gambar 10.7 Ubah file dalam zip



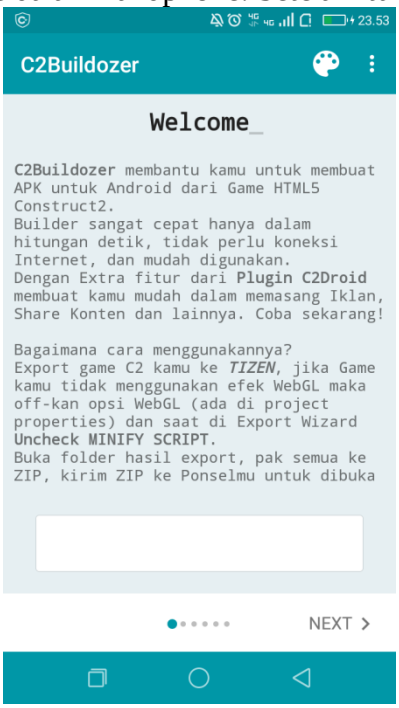
Gambar 10.8 Hasil file yang sudah di folder Zip

9. Selanjutnya copy folder zip kedalam handphone untuk selanjutnya kita compile dengan aplikasi C2 Buildozer.
10. Download aplikasi C2 Buildozer di playstore dan install aplikasi tersebut di handphone.



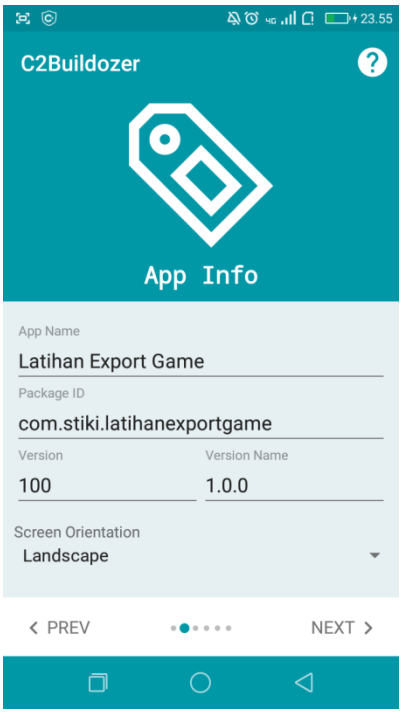
Gambar 10.9 Download aplikasi C2 Buildozer

11. Setelah C2 Buildozer terinstal bukalah aplikasi dan carilah folder zip export game yang sebelumnya di copy ke dalam handphone. Setelah itu klik next.



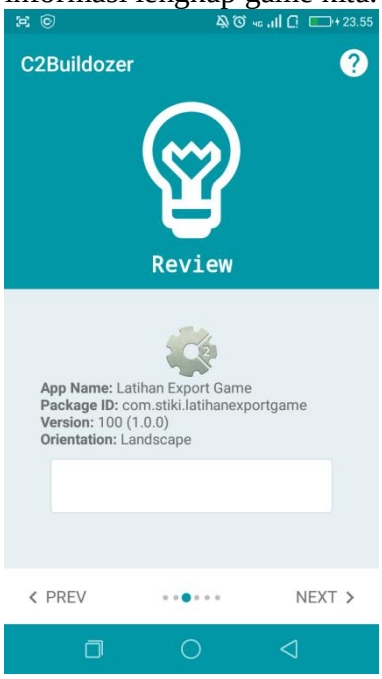
Gambar 10.10 Tampilan awal C2 Buildozer

12. Pada halaman ini akan terlihat data informasi game yang sebelumnya kita buat, selain itu di halaman ini kita bisa mengatur tampilan orientasi game landscape atau portrait. Setelah itu klik next.



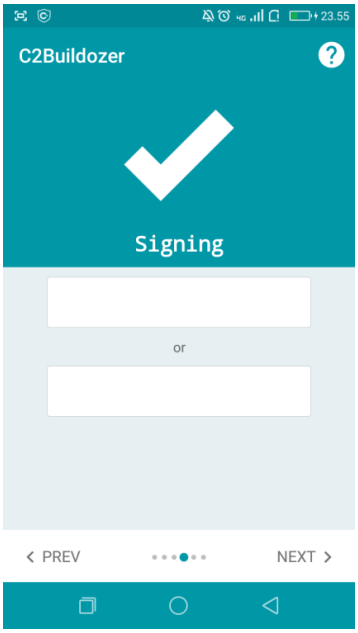
Gambar 10.11 Tampilan App Info pada C2 Buildozer

13. Pada halaman Review berisi informasi lengkap game kita.



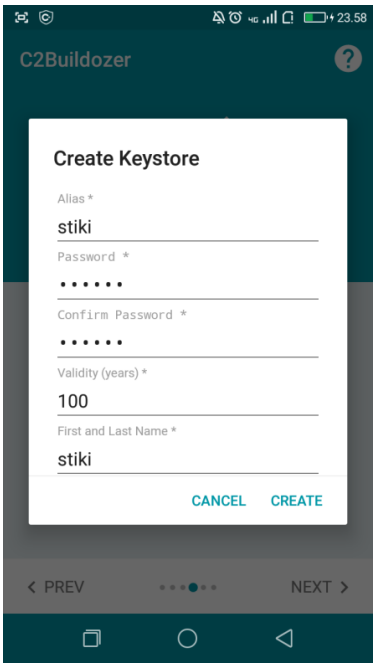
Gambar 10.12 Tampilan App Info pada C2 Buildozer

14. Pada halaman Signing kita bisa memvalidasi game kita dengan membuatkan keystore. Jika belum mempunyai keystore klik created keystore, namun jika sudah punya tinggal pilih keystore yang sudah terdaftar.



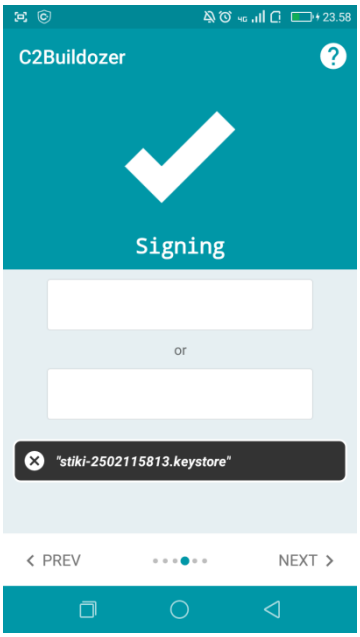
Gambar 10.13 Tampilan halaman signing pada C2 Buildozer

15. Pada halaman buat keystore isilah data dengan lengkap setelah itu klik create.



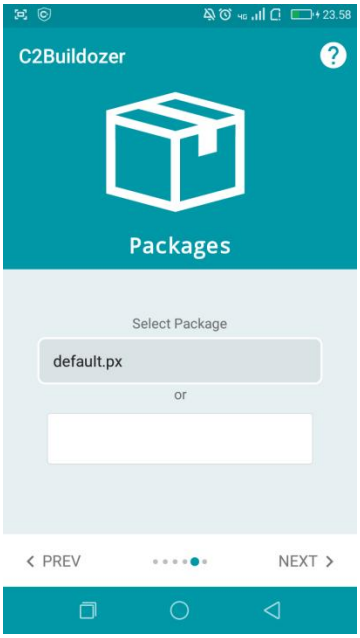
Gambar 10.14 Tampilan Create Keystore pada C2 Buildozer

Setelah itu klik next



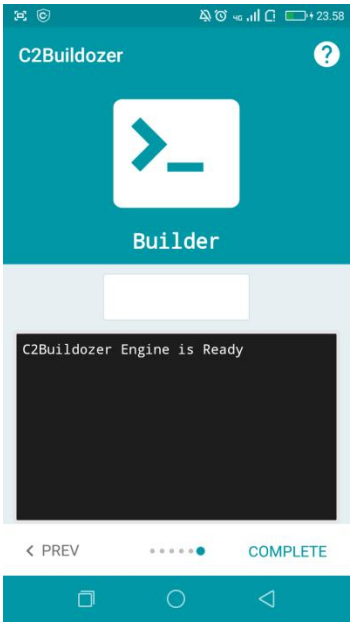
Gambar 10.15 Tampilan Keystore di pilih pada C2 Buildozer

16. Pada halaman packages klik next saja



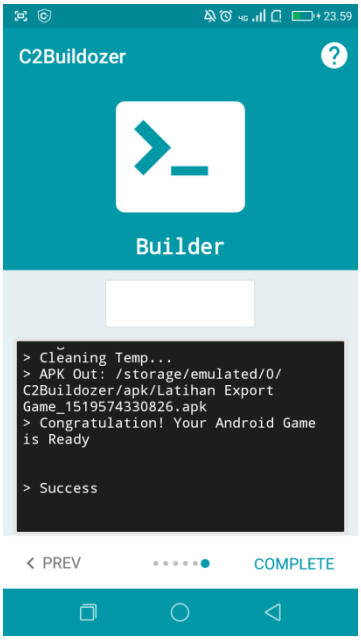
Gambar 10.16 Tampilan halaman packages pada C2 Buildozer

17. Setelah klik start build untuk memulai compile game yang kita buat ke dalam bentuk format .apk.



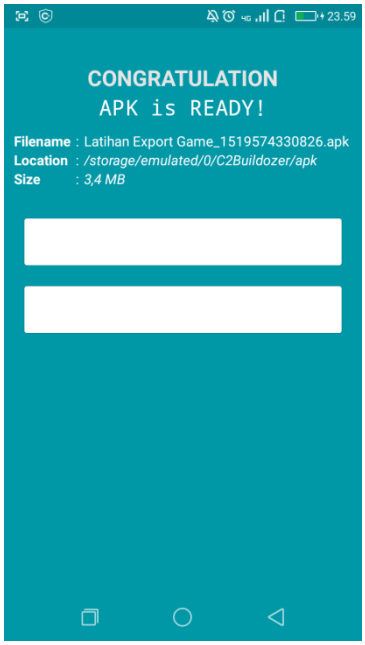
Gambar 10.17 Tampilan halaman builder pada C2 Buildozer

18. Setelah build selesai klik complete



Gambar 10.18 Tampilan selesai build pada C2 Buildozer

19. Game sudah berhasil di build menjadi format .apk. Pada halaman ini berisi informasi lokasi export apk game sudah di compile.



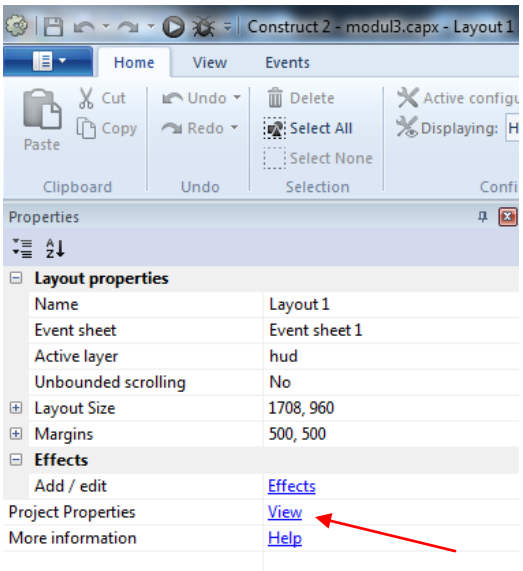
Gambar 10.19 Tampilan informasi lokasi export apk

KEGIATAN PRAKTIKUM 10.2
EXPORT GAME DENGAN COCOON IO

Pada praktikum 10.2 ini kita akan belajar cara mengexport game ke mobile secara online melalui web https://cocoon.io/.

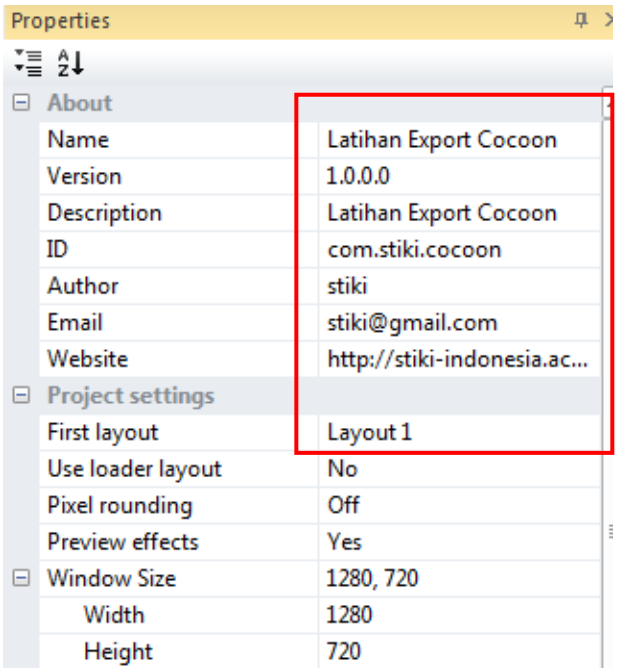
1. Pertama bukalah project game yang akan di export.

2. Klik view pada properties bar



Gambar 10.20 Export game dengan Cocoon Io langkah ke-1

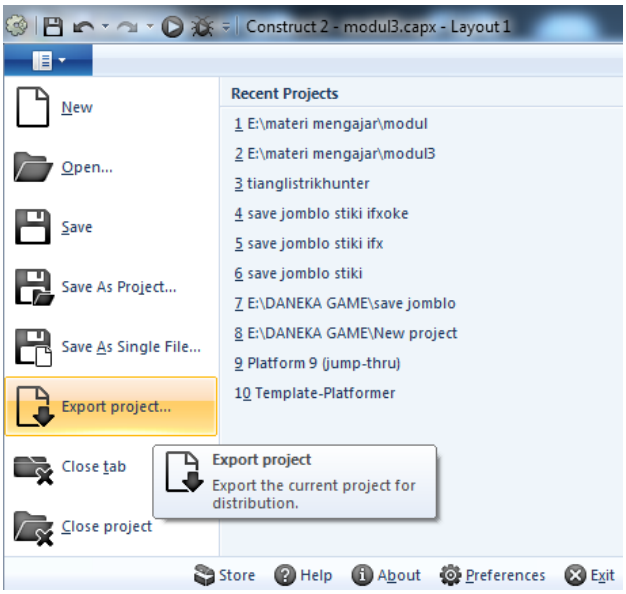
3. Isilah data about pada properties dengan lengkap seperti dibawah ini jangan sampai ada yang kosong. Pada bagian ID di harapkan defaultnya wajib diubah contoh "com.stiki.cocoon".



Gambar 10.21 Export game dengan Cocoon Io langkah ke-2

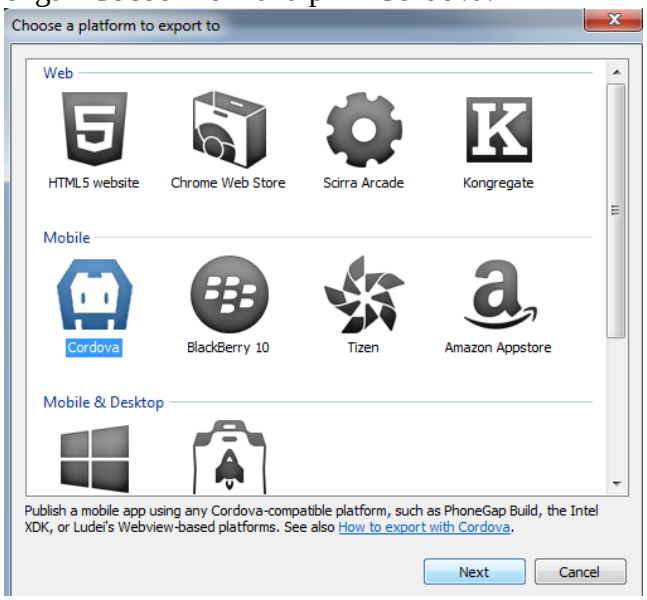
Pada bagian First Layout diharapkan di set sesuai layout apa yang akan kita play saat game mulai contoh kita set "Layout 1" sebagai layout awal yang kita play.

4. Setelah itu klik File > Export project



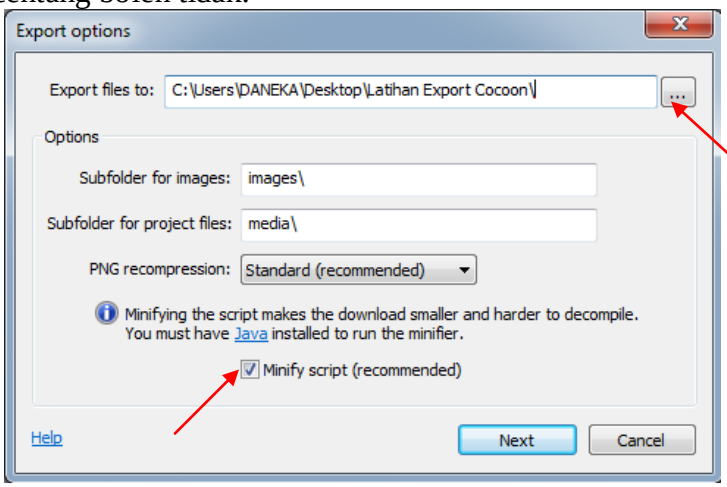
Gambar 10.22 Export game dengan Cocoon Io langkah ke-3

5. Pada tahap ini kita akan melihat banyak pilihan untuk export game, karena kita ingin export game kita dengan Cocoon Io maka pilih Cordova.



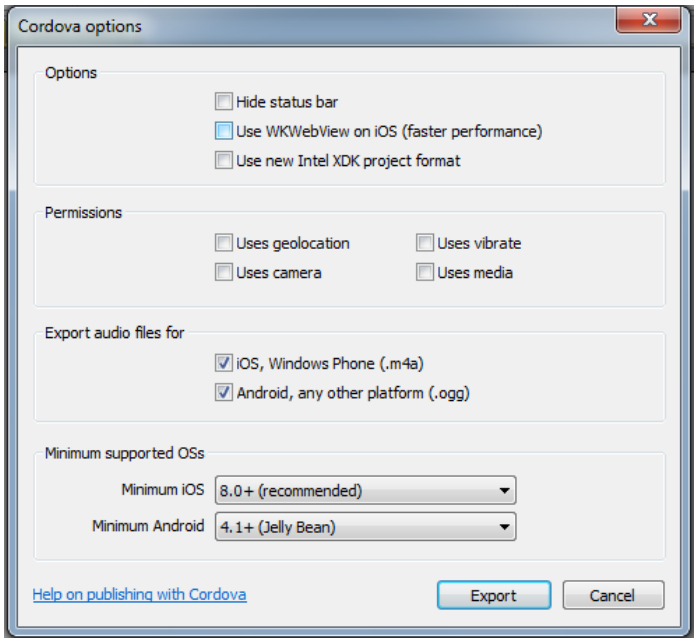
Gambar 10.23 Export game dengan Cocoon Io langkah ke-4

6. Dalam Export files to aturlah dimana game kita akan diexport dan bagian manify script boleh di centang boleh tidak.

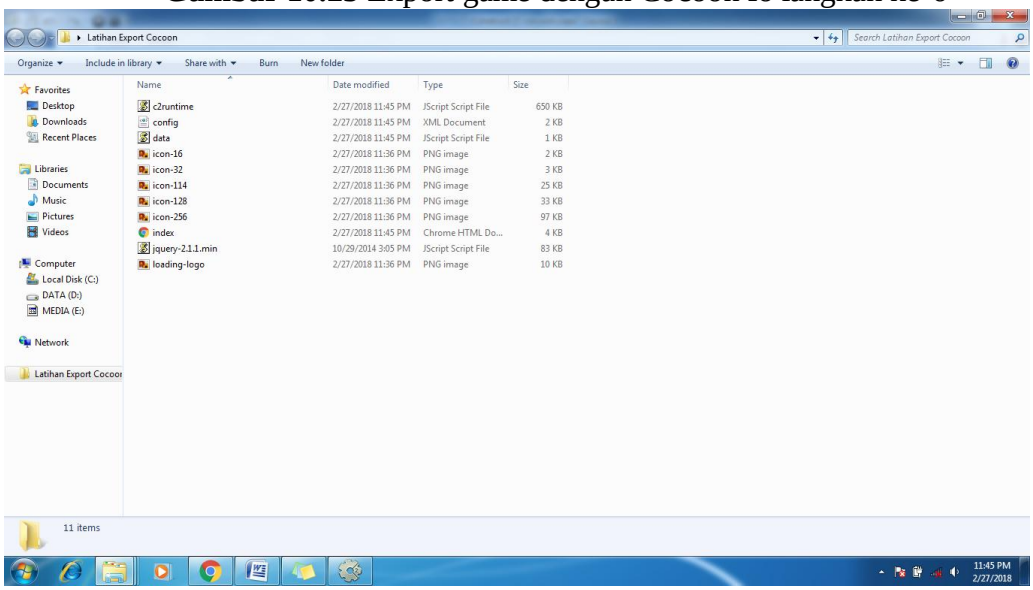


Gambar 10.24 Export game dengan Cocoon Io langkah ke-5

7. Pada halaman Cordova Option kita bisa mengatur untuk hasil exportnya. Setelah itu klik export dan tunggu beberapa saat sampai selesai game di compile.

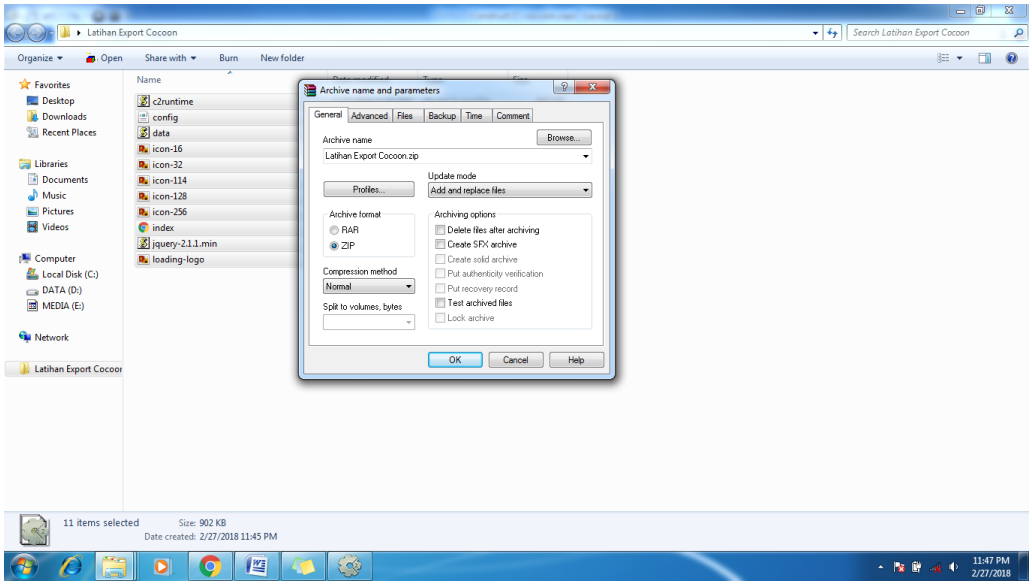


Gambar 10.25 Export game dengan Cocoon Io langkah ke-6

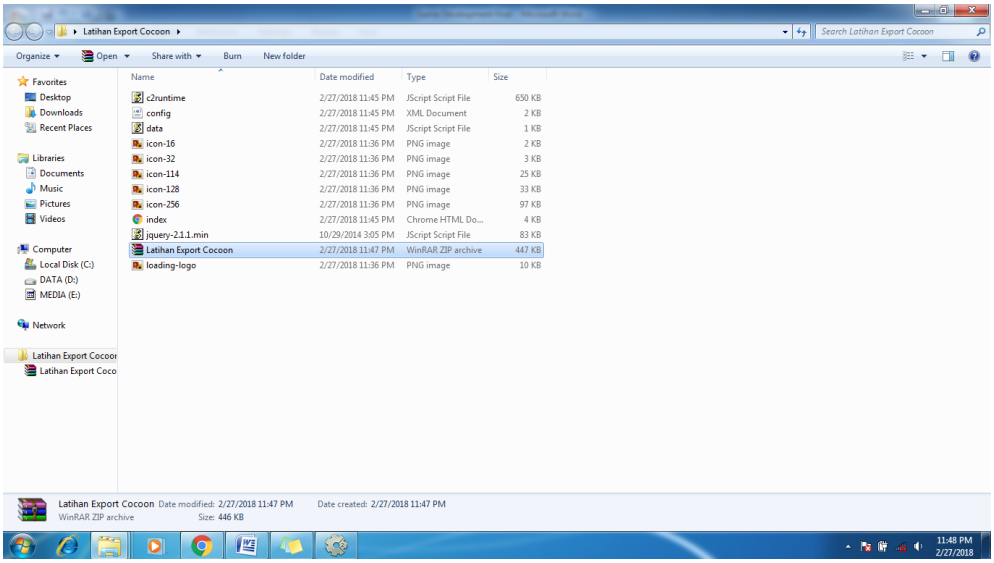


Gambar 10.26 Hasil Export

8. Saat game berhasil di compile selanjutnya select semua hasil exportnya dan jadikan dalam satu folder zip.

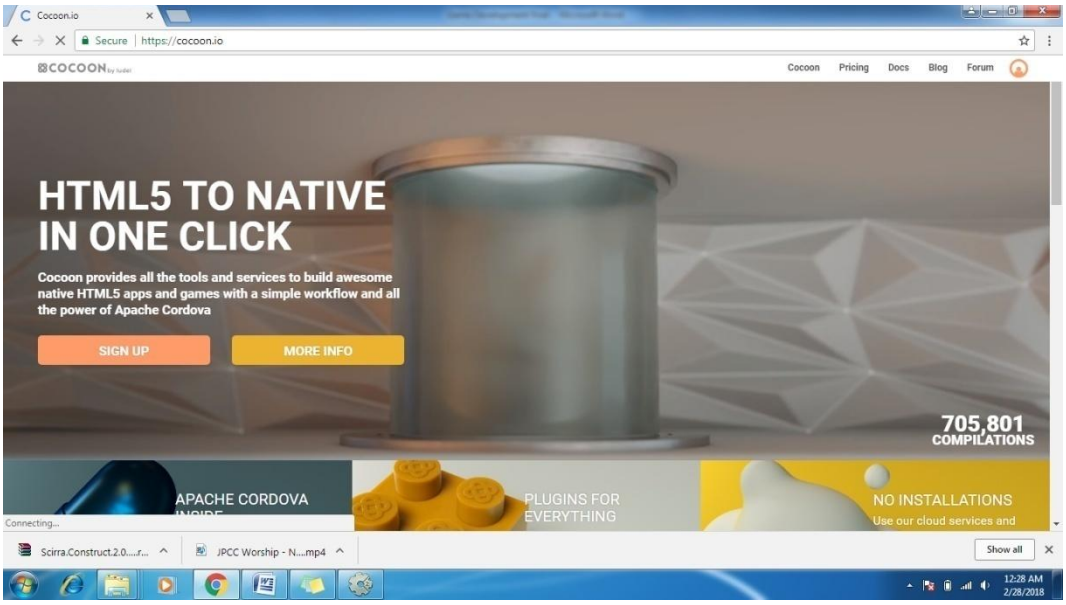


Gambar 10.27 Ubah file dalam zip



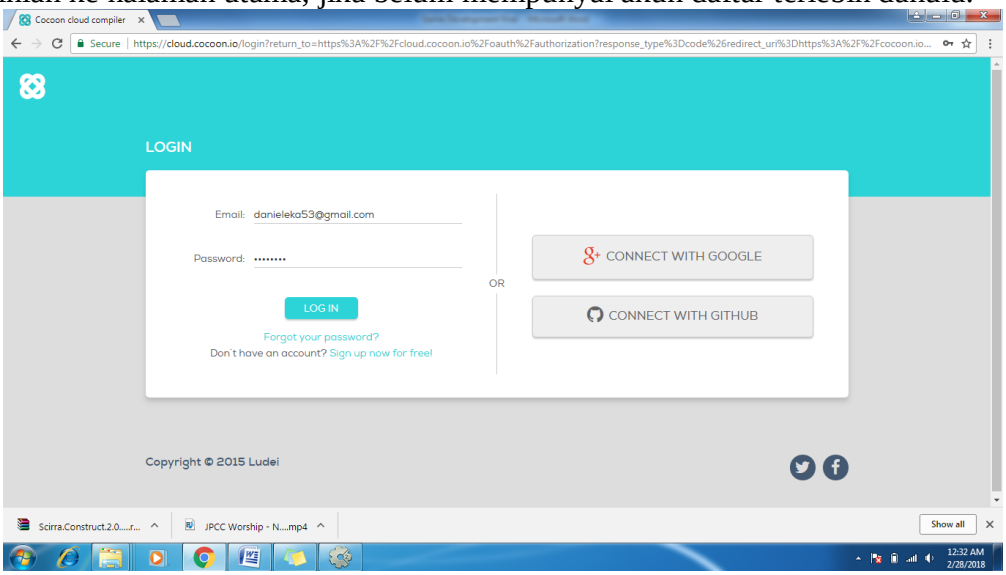
Gambar 10.28 Hasil file yang sudah di folder Zip

9. Bukalah web browser dan akses web https://cocoon.io/



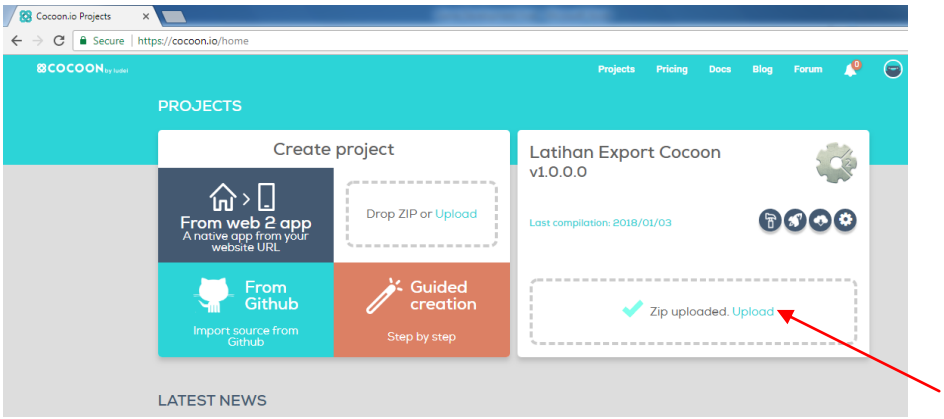
Gambar 10.29 Halaman awal Cocoon Io

10. Loginlah ke halaman utama, jika belum mempunyai akun daftar terlebih dahulu.

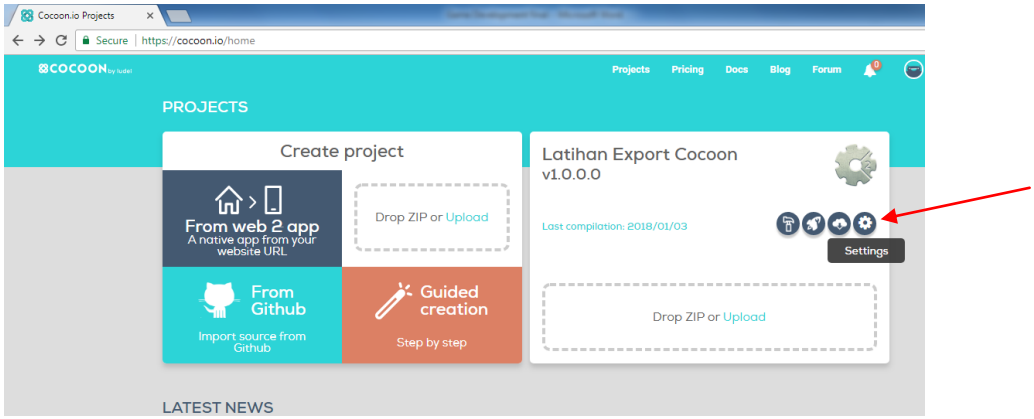


Gambar 10.30 Halaman login Cocoon Io

11. Klik upload atau Drag folder zip yang kita buat sebelumnya kedalam halaman Cocoon Io untuk mengimport game yang akan kita compile. Setelah mengimport klik tombol setting untuk mengatur ganme yang akan kita compile di Cocoon Io.

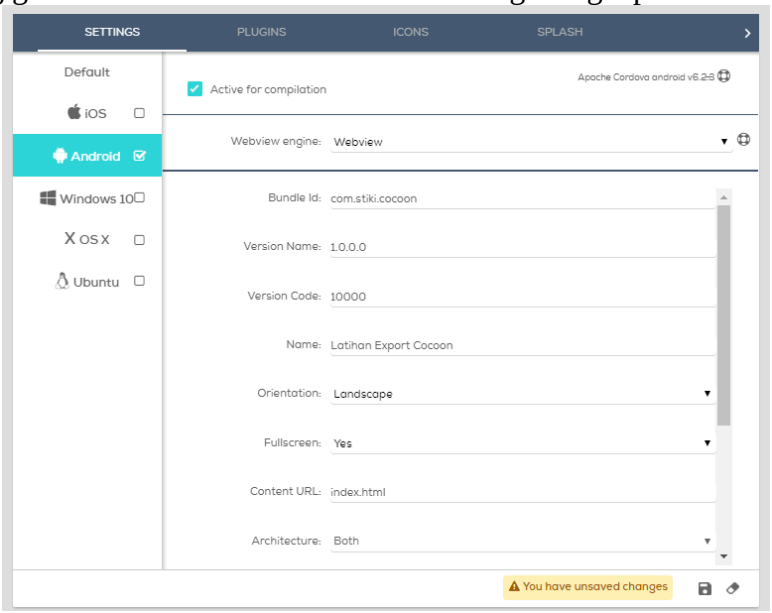


Gambar 10.31 Import zip ke Cocoon Io



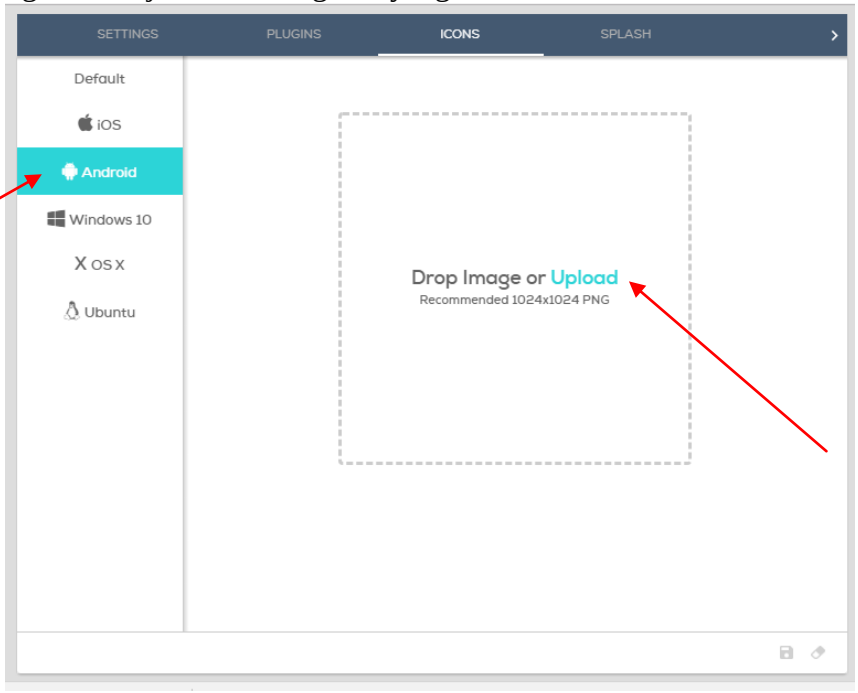
Gambar 10.32 Klik tombol Setting

12. Pada halaman setting isilah isian dengan lengkap seperti dibawah ini. Pada praktikum 10.2 kita akan mencoba meng-compile game kita ke dalam format .apk (android) maka centang gambar android. Setelah isian diisi denga lengkap klik save.

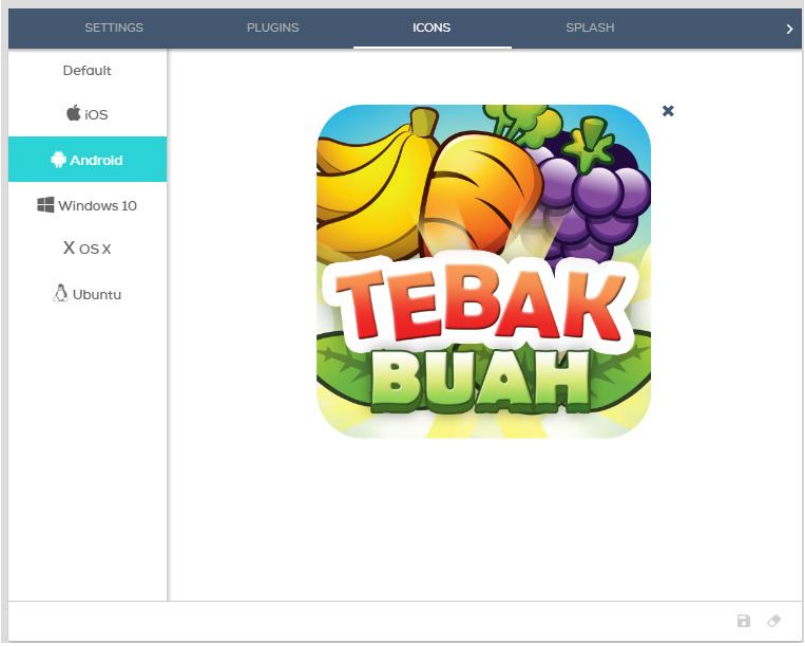


Gambar 10.33 Halaman setting Cocoon Io

13. Setelah itu klik halaman icon dan centang gambar android lalu klik upload atau drag gambar yang akan di jadikan icon game yang kita buat.

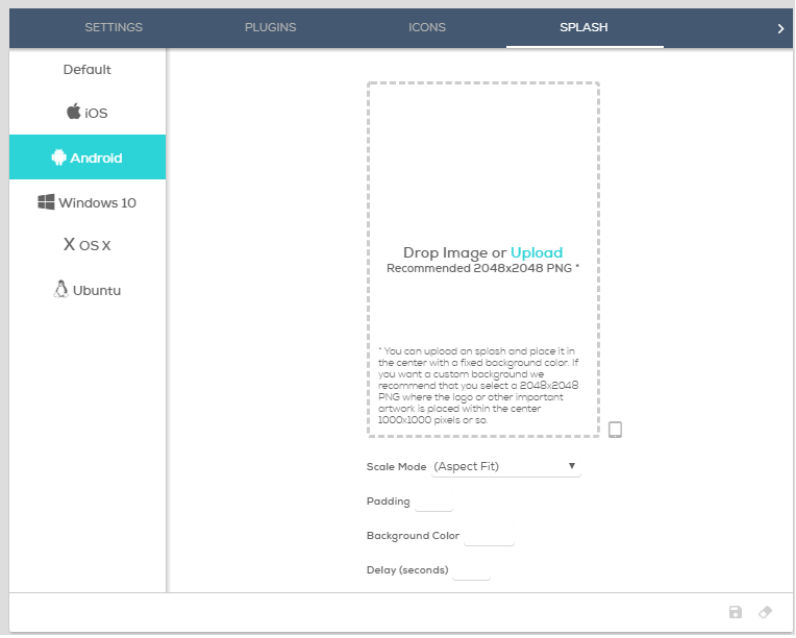


Gambar 10.34 Halaman Icons Cocoon Io



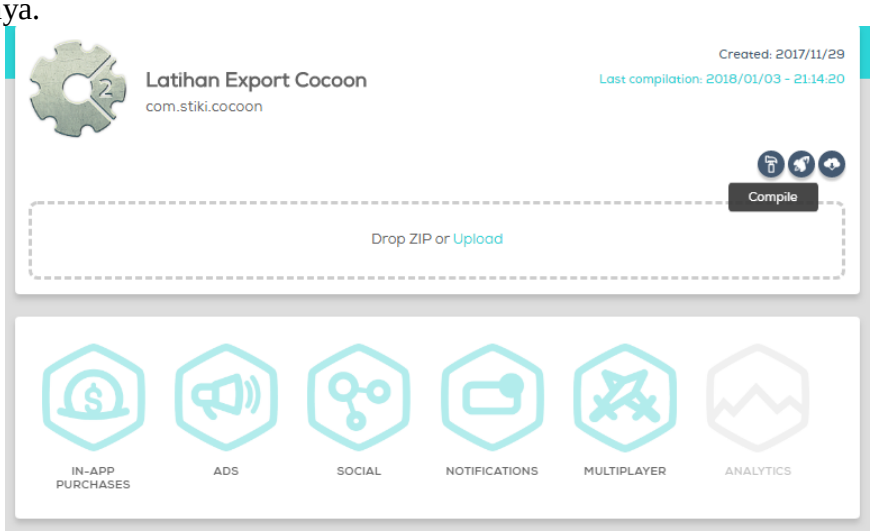
Gambar 10.35 Icon game yang di upload

14. Setelah itu klik halaman splash jika ingin membuat splash screen (tidak diharuskan).



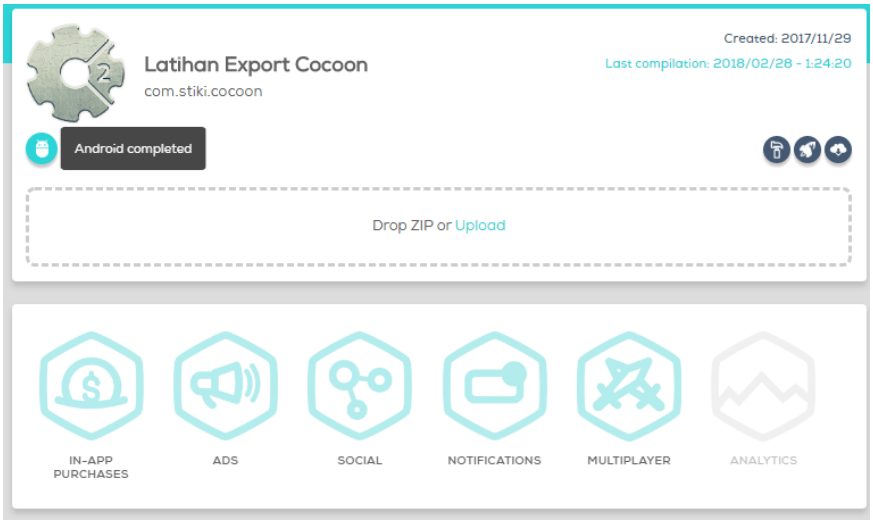
Gambar 10.36 Halaman Splash Cocoon Io

15. Setelah itu klik tombol compile untuk memulai compile game yang sudah kita setting sebelumnya.



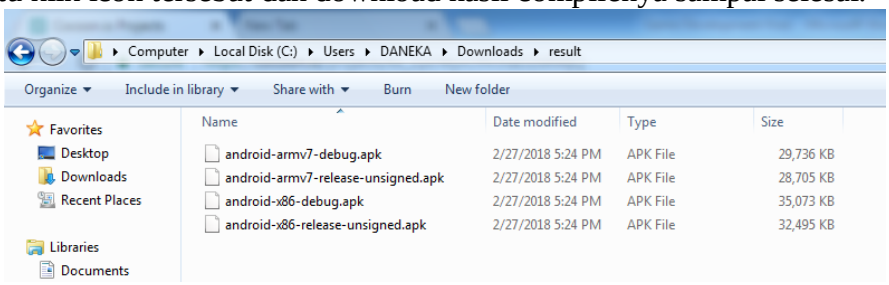
Gambar 10.37 Klik Tombol Compile

16. Setelah itu tunggulah compile game beberapa saat sampai icon berwarna hijau dan berstatus completed.



Gambar 10.38 Tampilan compile android completed

17. Setelah itu klik icon tersebut dan download hasil compilenya sampai selesai.



Gambar 10.39 Hasil compile android

TUGAS

1. Exportlah game yang kalian buat menggunakan C2 Buildozer dan Cocoon Io dan bandingkan hasilnya ! Sebutkan apa perbedaan yang kalian dapatkan !

MODUL XI

TIPS & TRICKS

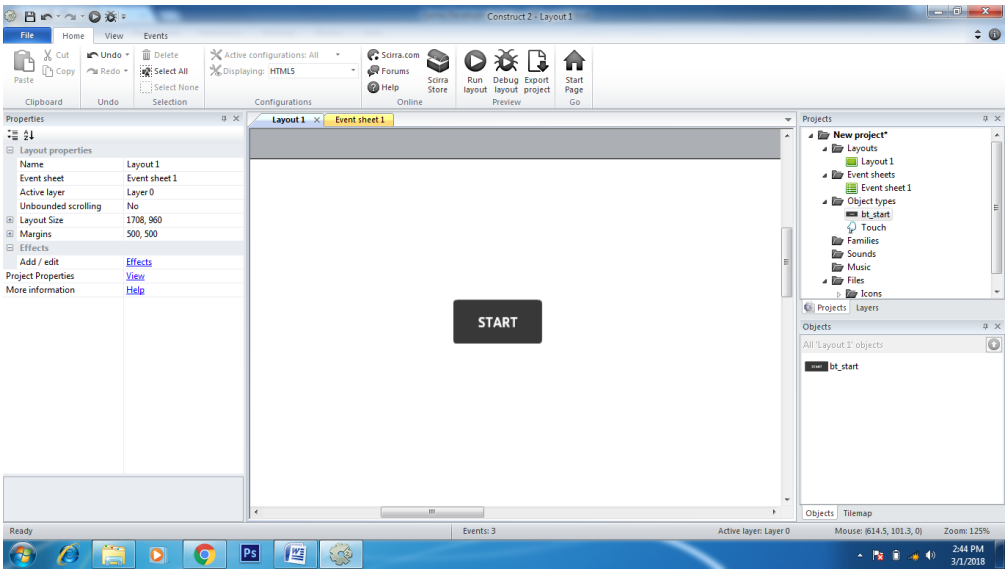
(Pertemuan 13-15)

11.1 Button Animations

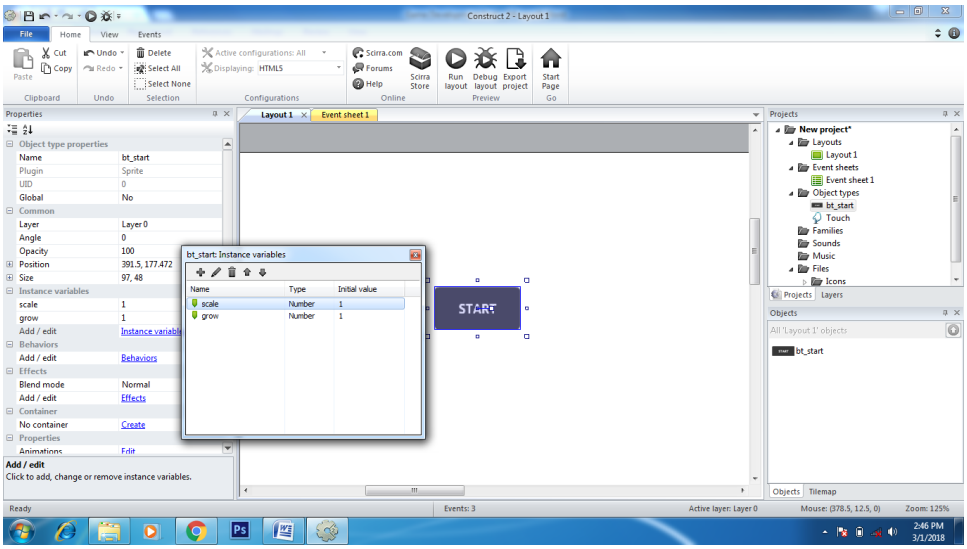
Untuk membuat tampilan menu game terkesan lebih hidup, ada baiknya kita membuat tombol-tombol di dalamnya memiliki animasi. Kita akan membuat animasi tombol sederhana yang akan membesar dan mengecil saat di sentuh.

1. Hal terpenting yang kita butuhkan adalah input touch untuk menjadi trigger menjalankan animasi. Selain itu, juga dapat menggunakan input keyboard maupun mouse tergantung sasaran game yang diinginkan.

2. Masukan asset tombol start pada layout beri nama **"bt_start"**. Lalu berikan instance variable angka bernama **"scale"** dan **"grow"**, beri initial value 1.



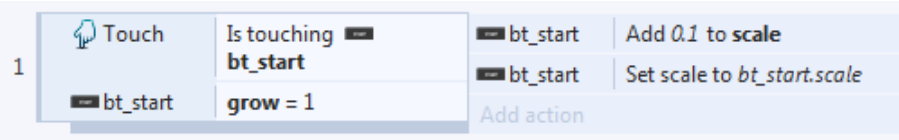
Gambar 11.1 Memasukan tombol start pada layout



Gambar 11.2 Tambah instance variable scale dan grow pada bt_start

3. Pertama kita akan membuat tombol bertambah besar jika dipilih. Tambahkan kode ini pada event sheet.

- **Add event > Touch > Is touching object > button**
- **Add another condition > bt_start > Compare instance variable > grow = 1**
  - **Add action > bt_start > Add value > 0.1 to scale**
  - **Add action > bt_start > Set scale > bt_start.scale**

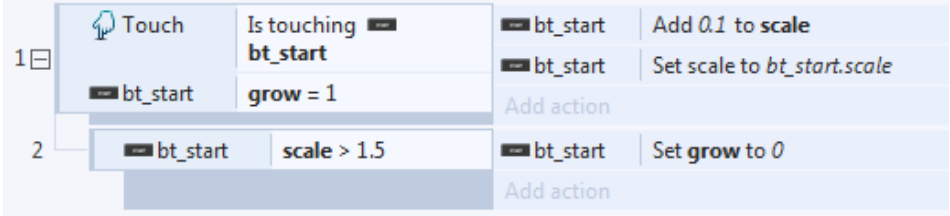


Gambar 11.3 Event sheet bt_start membesar saat di sentuh

Nilai 0.1 menandakan seberapa cepat waktu yang di perlukan untuk bertambah besar. Makin kecil nilainya, maka akan makin lama.

4. Jika melakukan playtest makin lama tombol di tekan, maka tombol akan makin bertambah besar. Untuk mengatasinya, kita batasi sampai seberapa besar tombol tersebut tumbuh. Tambahkan kode ini pada event sheet.

- **Add sub-event > bt_start > Compare instance variable > scale > 1.5**
  - **Add action > bt_start > Set value > set grow to 0**

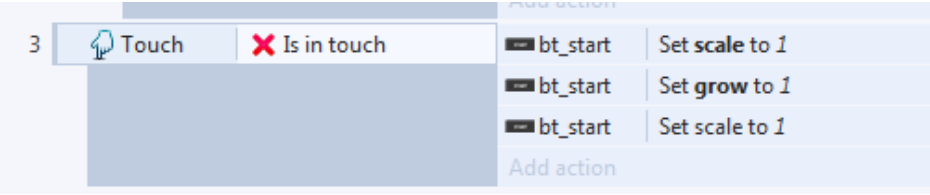


Gambar 11.4 Event sheet batas pembesaran bt_start

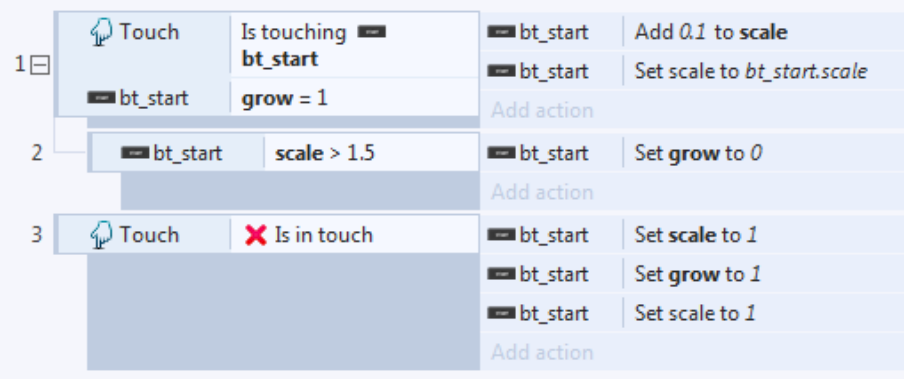
Nilai 1.5 menandakan seberapa batas besar tombol saat di tekan. Makin besar nilainya maka semakin besar ukuran tombol saat di tekan.

5. Sekarang pertumbuhan tombol sudah tidak liar seperti tadi, akan tetapi muncul masalah baru, yaitu tombol tidak bisa mengecil kembali. Maka tambahkan kode pada event sheet kita untuk set tombol kembali ke ukuran semula jika tidak disentuh.

- **Add event > Touch > Is in touch > Invert**
  - **Add action > bt_start > Set value > Set scale to 1**
  - **Add action > bt_start > Set value > Set grow to 1**
  - **Add action > bt_start > Set scale > 1**



Gambar 11.5 Event sheet saat bt_start tidak di sentuh

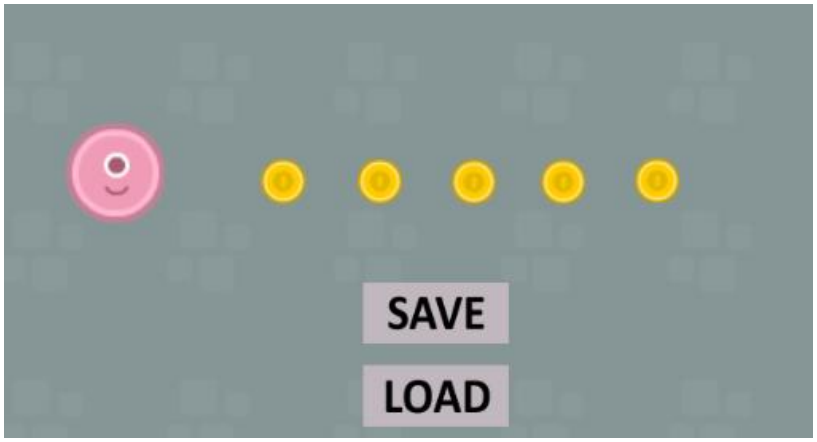


Gambar 11.5 Event sheet seluruhnya button animation

11.2 Animated Menu

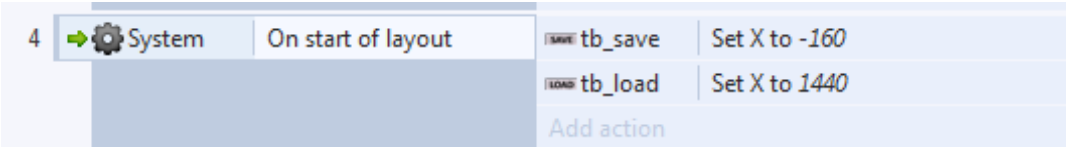
Saat kalian bermain game pastinya pernah melihat game saat di menu utama tombol-tombolnya bisa bergerak baik dari atas ke bawah maupun dari kanan ke kiri. Tampilan menu seperti ini sangatlah menarik sehingga membuat tampilan game tersebut lebih hidup. Kita akan membuat tampilan animasi tombol menu sedeharna agar game kita lebih menarik.

1. Pertama kita bisa membuka project game yang sebelumnya kita buat pada modul IX praktikum 9.1. Kita akan membuat tombol menu Save dan Load tersebut saat game di mulai bisa bergerak dari kanan ke kiri maupun dari kiri ke kanan.



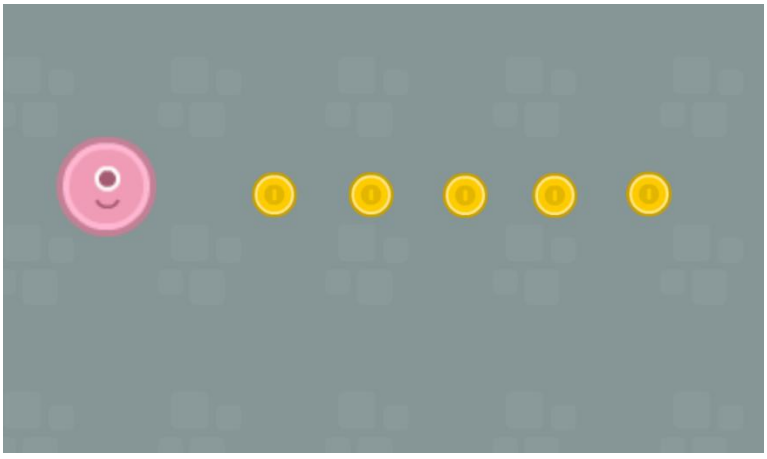
Gambar 11.6 Membuka project game praktikum 9.1

2. Selanjutnya kita tambahkan kode pada event sheet kita untuk memposisikan tombol Save dan Load pada saat game di mulai berada di luar layout. Pada project game yang sebelumnya kita buat ukuran layoutnya yaitu 1280 x 720 px, maka agar tombol tersebut tidak terlihat saat game di mulai di dalam layout usahakan posisikan tombol melebihi ukuran layout. Disini kita akan posisikan tombol Save X= - 160 dan tombol Load = X 1440.



Gambar 11.7 Event sheet memposisikan tombol saat game di mulai

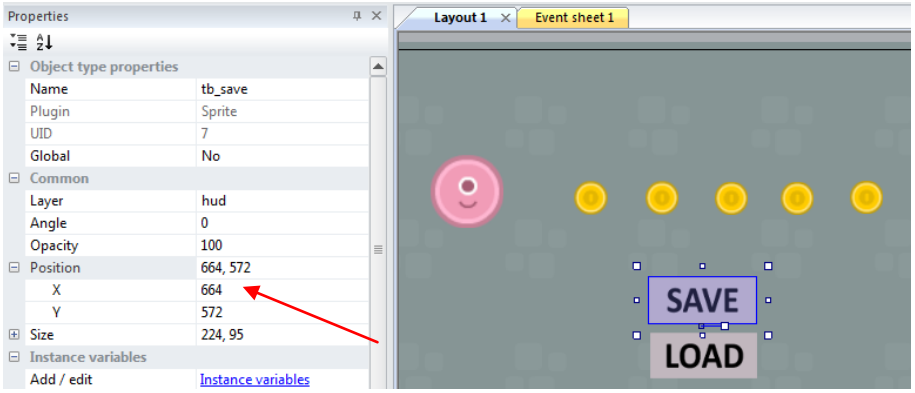
3. Setelah itu cobalah playtest untuk melihat apakah posisi tombol Save dan Load sudah berada di luar layout sehingga tidak terlihat.



Gambar 11.8 Tampilan playtest tombol yang sudah di luar layout

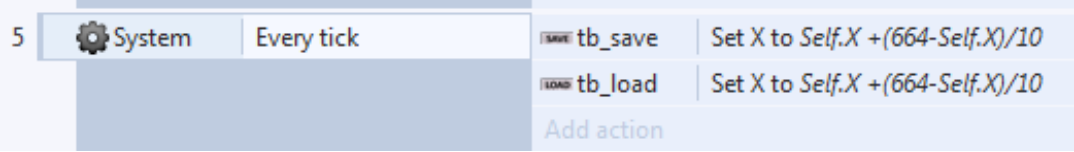
4. Kita sudah berhasil membuat tombol Save dan Load berada di luar layout. Selanjutnya kita buat tombol Save dan Load bergerak ke posisi yang di tentukan. Sebelum kita

memasukan kode pada event sheet kita catat dulu posisi kordinat X tombol Save dan Load dengan cara klik tombol dan lihat pada properties berapa nilai position. Pada project sebelumnya posisi kordinat X tombol Save dan Load kebetulan berada pada posisi kordinat **X=664.**



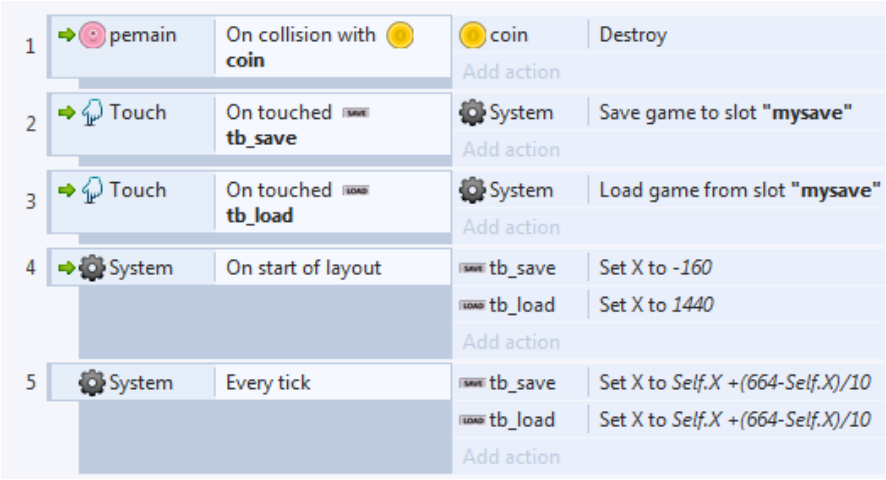
Gambar 11.9 Mengecek posisi kordinat X pada tombol Save

5. Setelah mengetahui posisi kordinatnya selanjutnya kita tambahkan kode untuk menggerakan tombol Save dan Load ke posisi kordinat yang di tentukan.



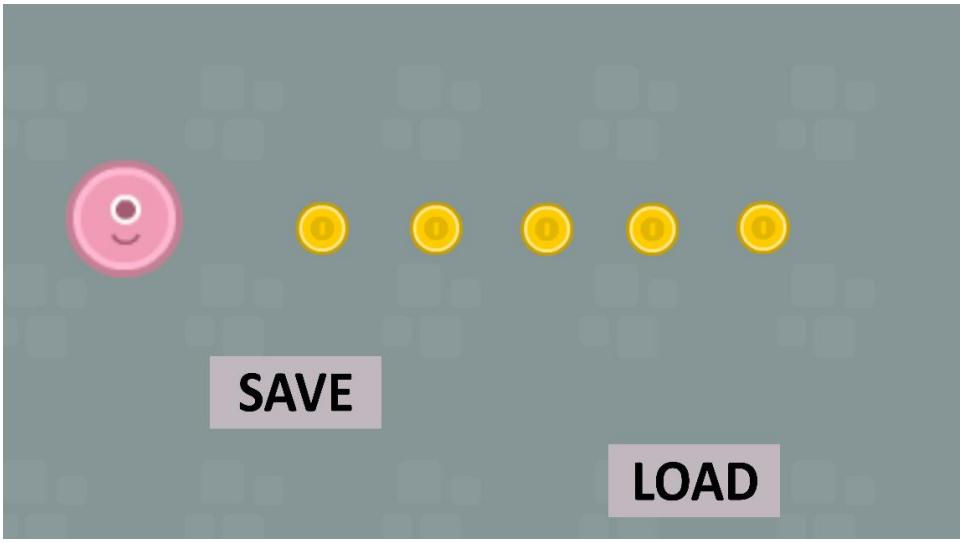
Gambar 11.10 Event sheet menggerakan tombol ke posisi kordinat X=664

Nilai 10 pada event sheet diatas adalah untuk mengatur seberapa lama untuk sampai ke posisi kordinat yang ditentukan. Semakin besar nilainya maka akan semakin lama untuk mencapai posisi kordinat yang ditentukan.

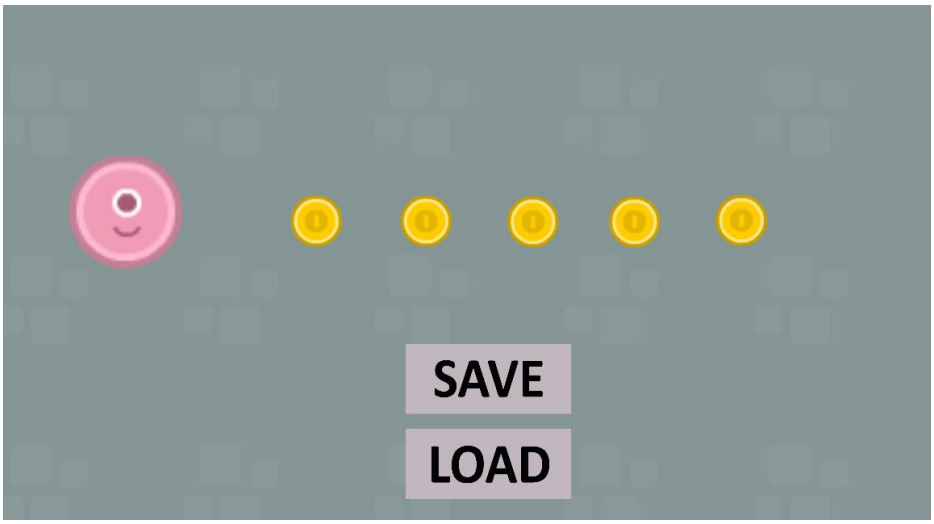


Gambar 11.11 Event sheet seluruhnya Animated Menu

6. Setelah itu cobalah playtest dan lihat hasilnya.



Gambar 11.12 Tampilan playtest tombol bergerak menuju kordinat yang di tentukan

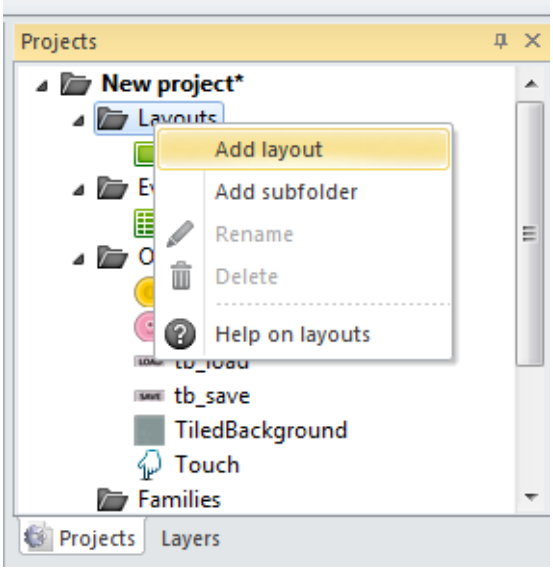


Gambar 11.13 Tampilan playtest tombol sampai pada kordinat yang di tentukan

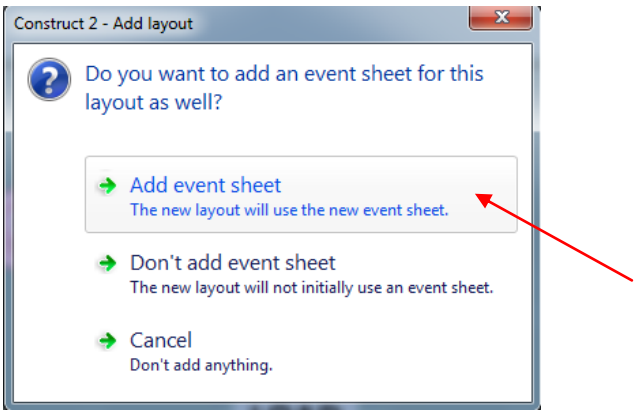
11.3 Splash Screen

Saat kalian bermain game pastinya pernah melihat Splash Screen yang menampilkan Logo Game Developer yang membuat game tersebut sebelum menuju halaman menu. Kita akan membuat Splash Screen sebagai intro awal sebelum menuju halaman menu game kita.

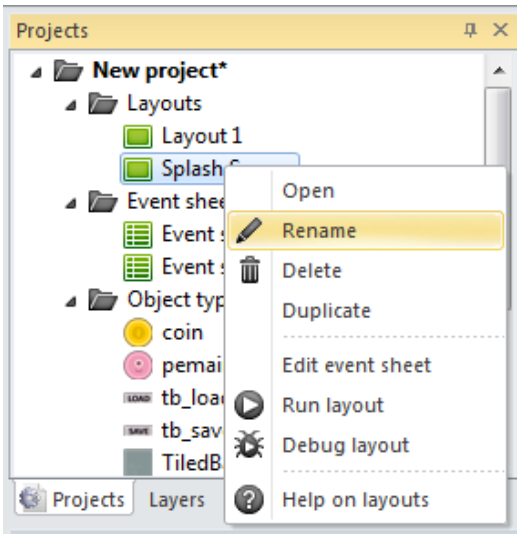
1. Pertama kita bisa membuka project game yang sebelumnya kita buat pada praktikum 9.2. Kita akan membuat halaman Splash Screen sebelum menuju halaman game.
2. Selanjutnya kita tambahkan Layout baru dengan cara **klik kanan > Add layout** lalu pada opsi tambah layout pilih **Add event sheet** setelah itu beri nama **"Splash Screen"** dengan cara **klik kanan layout > Rename**.



Gambar 11.14 Tambah Layout

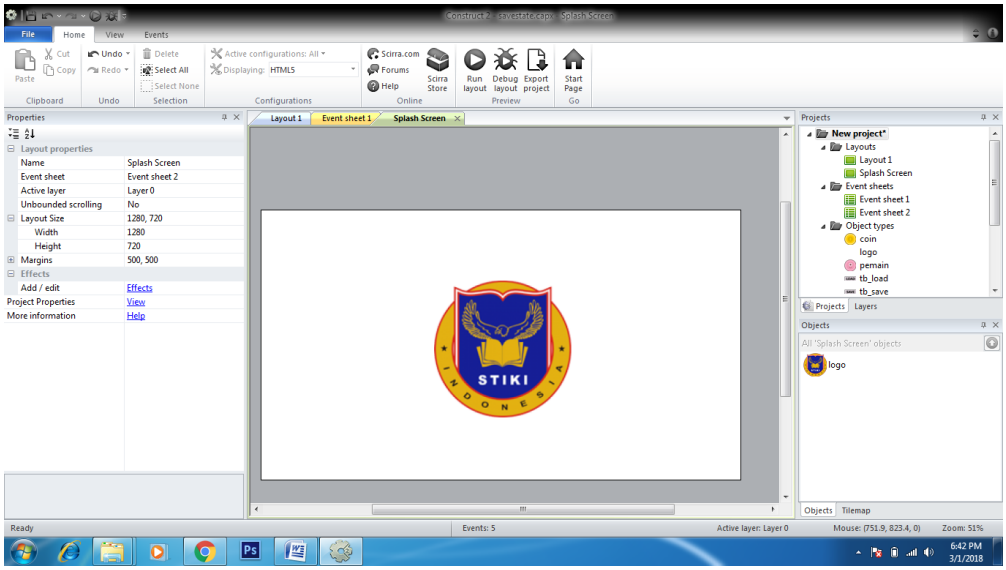


Gambar 11.15 Opsi Tambah Layout



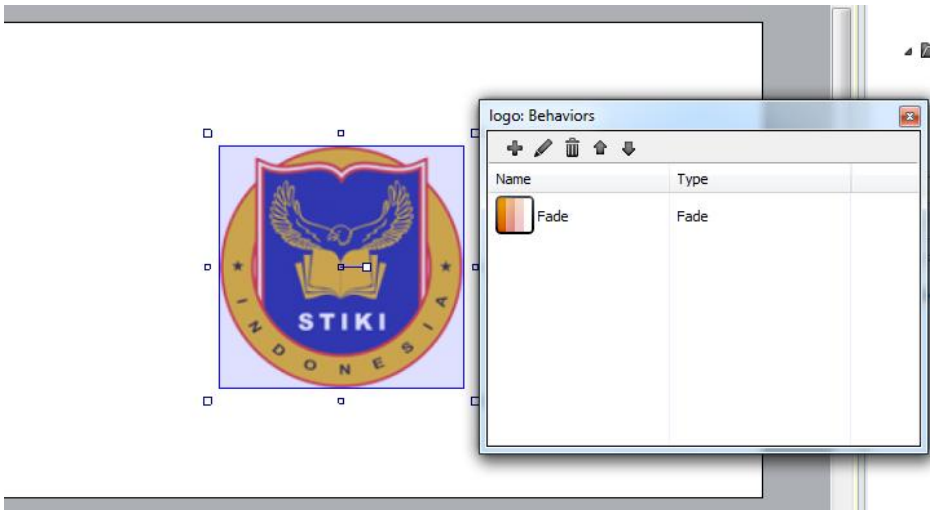
Gambar 11.16 Ubah nama layout menjadi Splash Screen

3. Selanjutnya tambahkan logo kedalam Layout Splash Screen dan beri nama logo.



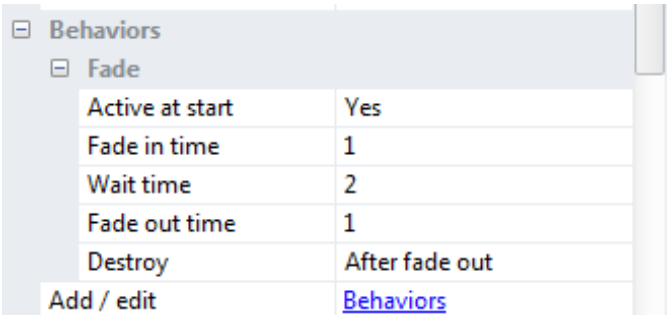
Gambar 11.17 Tambah logo pada layout Splash Screen

4. Setelah itu klik object logo tambahkan behavior Fade.



Gambar 11.18 Tambah behavior Fade pada logo

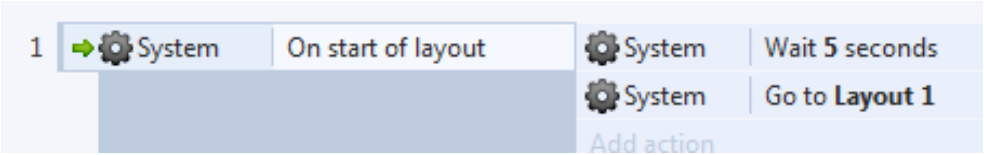
5. Selanjutnya aturlah bagian properties Behavior Fade seperti dibawah ini



Gambar 11.19 Setting properties Behavior Fade

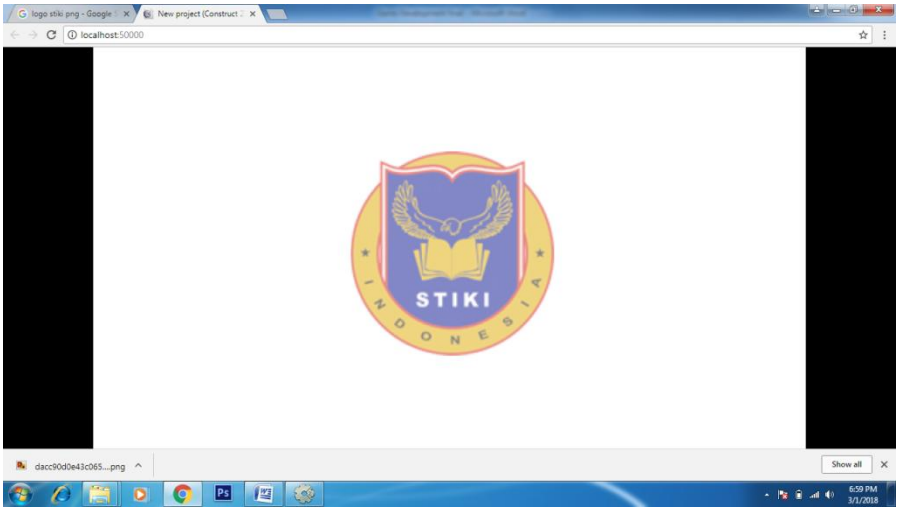
Pengaturan di atas Fade in time untuk mengatur waktu muncul logo selama 1 detik, Wait time untuk mengatur waktu di tampilkannya logo selama 2 detik, dan Fade out time untuk mengatur waktu menghilangnya logo selama 1 detik.

6. Tambahkan kode untuk mengatur kondisi saat layout di Start tunggu selama 5 detik lalu pindah ke Layout 1 pada event sheet.

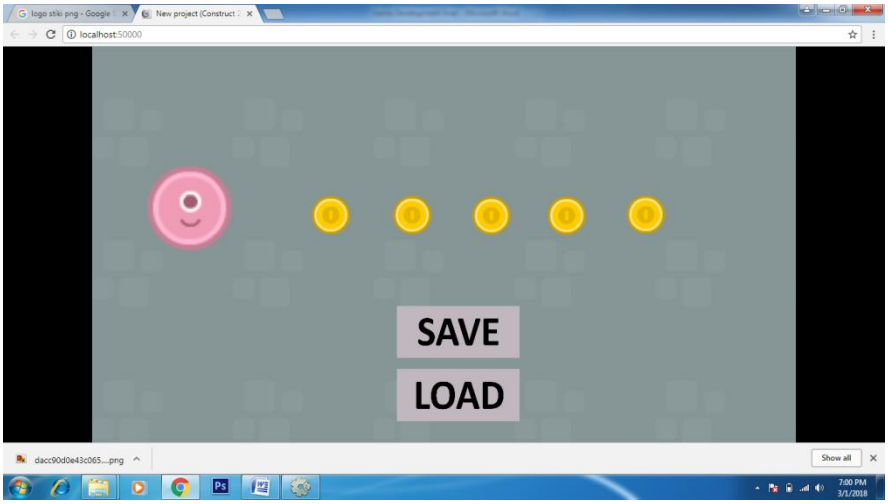


Gambar 11.20 Event sheet pindah Layout 1

7. Setelah itu cobalah playtest maka logo akan muncul dan menghilang hingga akhirnya pindah ke Layout 1 pada detik ke 5.



Gambar 11.21 Tampilan playtest Fade In Logo



Gambar 11.22 Tampilan playtest saat pindah ke Layout 1

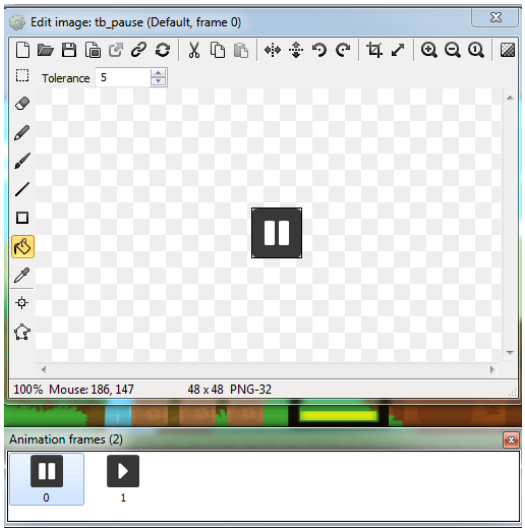
11.4 Pause Menu

Seperti yang kita tahu, Pause Menu berfungsi untuk menghentikan game untuk sementara waktu. Hasil dari teknik ini adalah membuat semua objek di dalam game menjadi freeze alias tidak bisa bergerak.

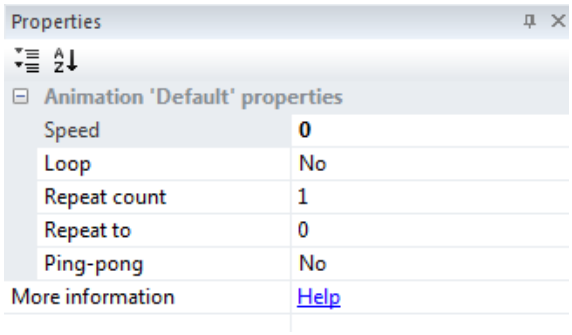
1. Pertama kita bisa membuka project game yang sebelumnya kita buat pada praktikum 7.2. Kita akan membuat tombol pause menu pada game tersebut.

Gambar 11.23 Membuka project game praktikum 7.2

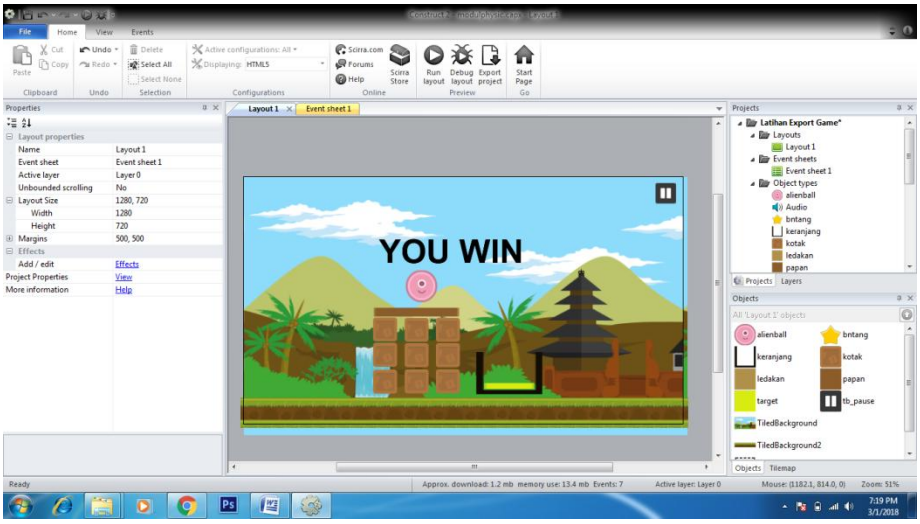
2. Selanjutnya tambahkan tombol pause dan beri nama **"tb_pause"**, yang dimana mempunyai animasi yang berisi 2 frame dengan gambar pause dan play. Lalu set Speednya menjadi 0 pada properties animation agar tidak bergerak Animation frame tb_pause saat di play.



Gambar 11.24 Tambah Animation frame dalam tb_pause

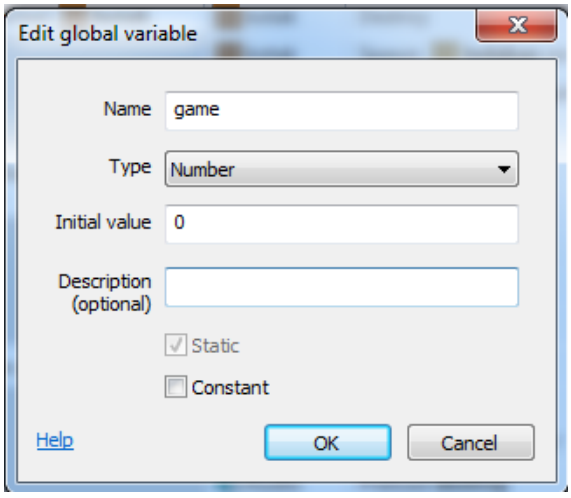


Gambar 11.25 Set Speed properties Animation



Gambar 11.25 Tampilan Desain Game Menu Pause

3. Selanjutnya pada event sheet kita buatkan global variable dengan nama **"game"** dengan nilai 0.

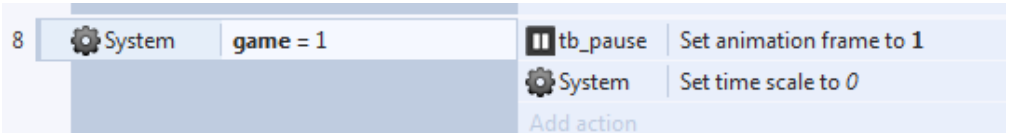


Gambar 11.26 Tambah global variable dengan nama "game"

4. Selanjutnya kita tambahkan kode yang dimana berisi kondisi saat tb_pause di tekan maka tambahkan nilai variable “game” dengan nilai 1.

Gambar 11.27 Event sheet saat tb_pause di tekan

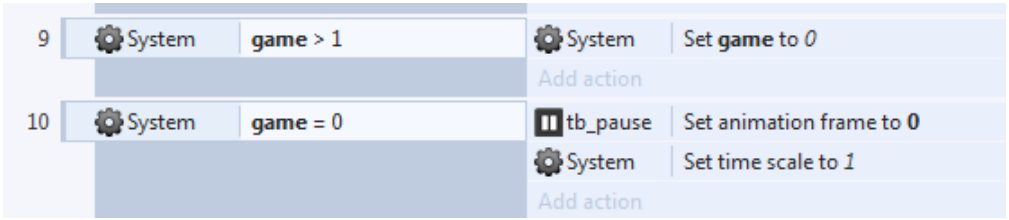
5. Selanjutnya buatlah kondisi saat variable game bernilai 1 maka pause game dan set ke frame 1 untuk mengganti gambar pause dengan gambar play.



Gambar 11.28 Event sheet pause game

Set time scale bernilai 0 artinya dimana semua object dalam game akan di pause sehingga tidak bergerak.

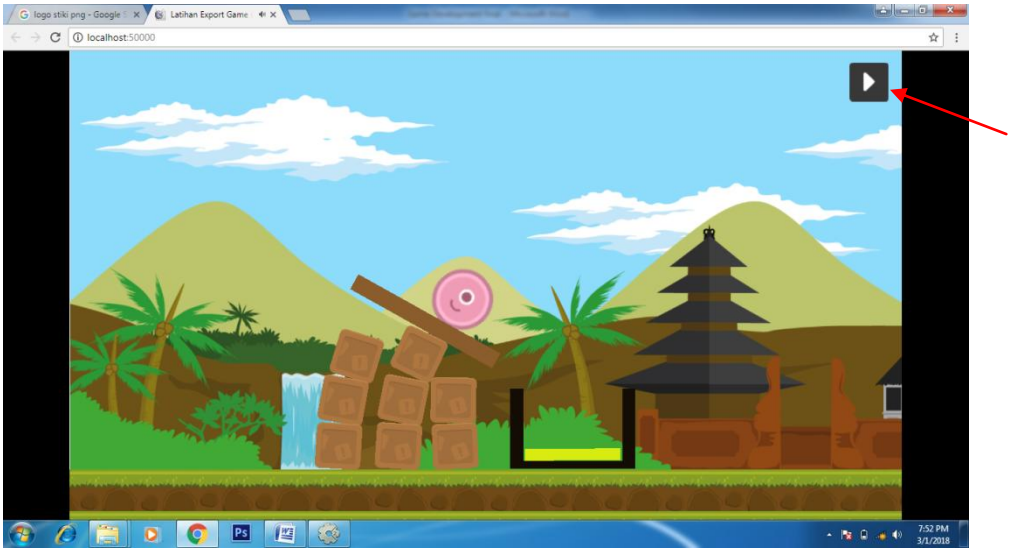
6. Selanjutnya tambahkan kondisi jika variable game bernilai lebih dari 1 maka set nilai variable game dengan 0 agar object dalam game bisa kembali berjalan seperti semula.



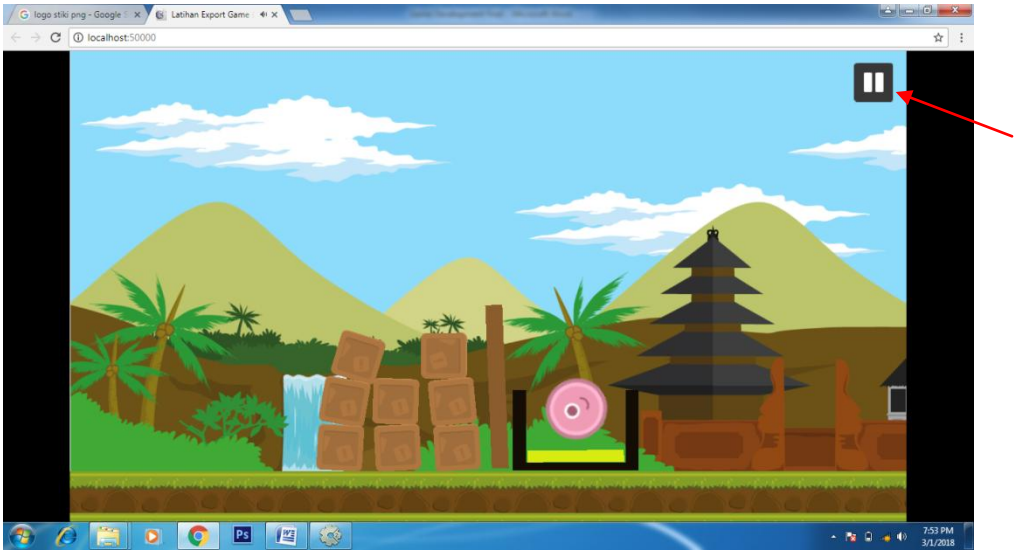
Gambar 11.29 Event sheet play game

Set time scale bernilai 1 artinya dimana semua object dalam game akan bisa bergerak kembali seperti semula.

7. Selanjutnya lakukan playtest dan cobalah tekan tb_pause untuk mencoba apakah game sudah bisa pause dan play kembali.



Gambar 11.30 Tampilan playtest game di pause

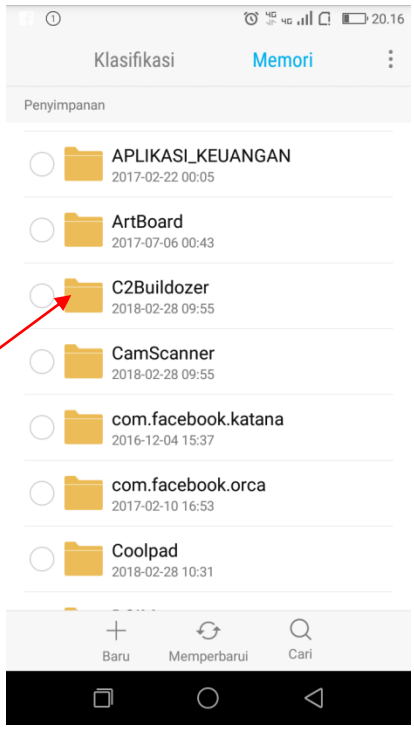


Gambar 11.31 Tampilan playtest game tidak di pause

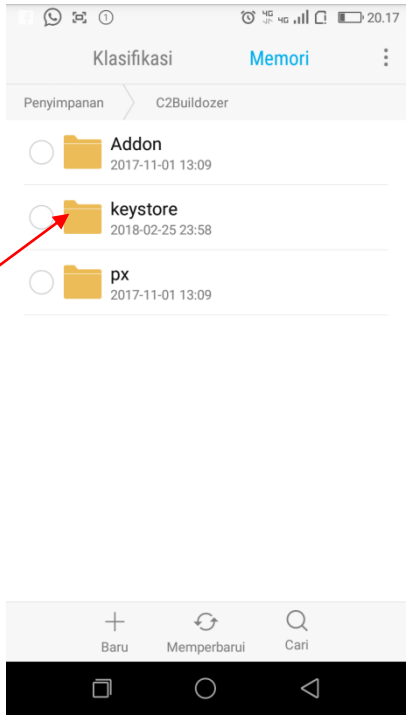
11.5 Keystore

Keystore adalah tanda tangan digital pembuat aplikasi. Kalau tidak di tanda tangani maka aplikasi atau game yang kita buat tidak akan bisa di upload ke Android market. Pada praktikum 10.2 sebelumnya kita sudah meng-compile game melalui Cocoon Io, namun game yang kita buat belum di tanda tangani atau di berikan Keystore, sehingga hasil compilenya menjadi banyak dan tentu saja tidak bisa diupload ke Android market.

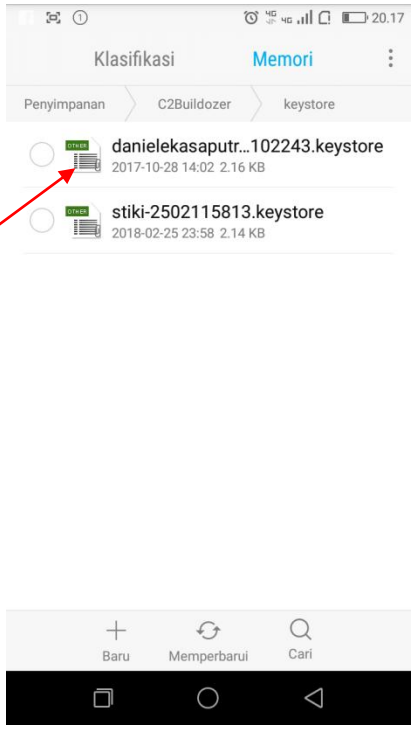
1. Pertama kita buka tempat penyimpanan keystore yang sebelumnya kita buat menggunakan C2 Buildozer di dalam handphone kita.



Gambar 11.32 Langkah mencari file keystore ke-1

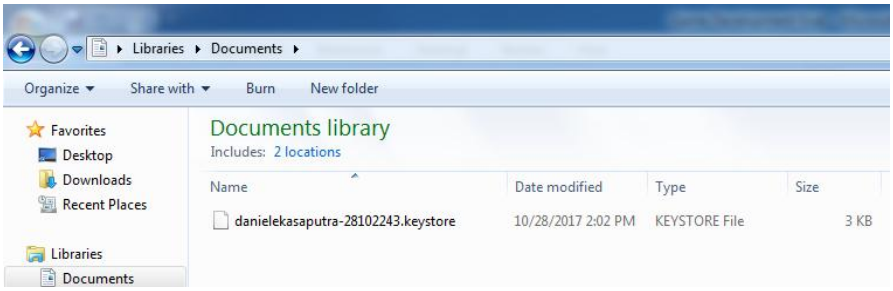


Gambar 11.33 Langkah mencari file keystore ke-2



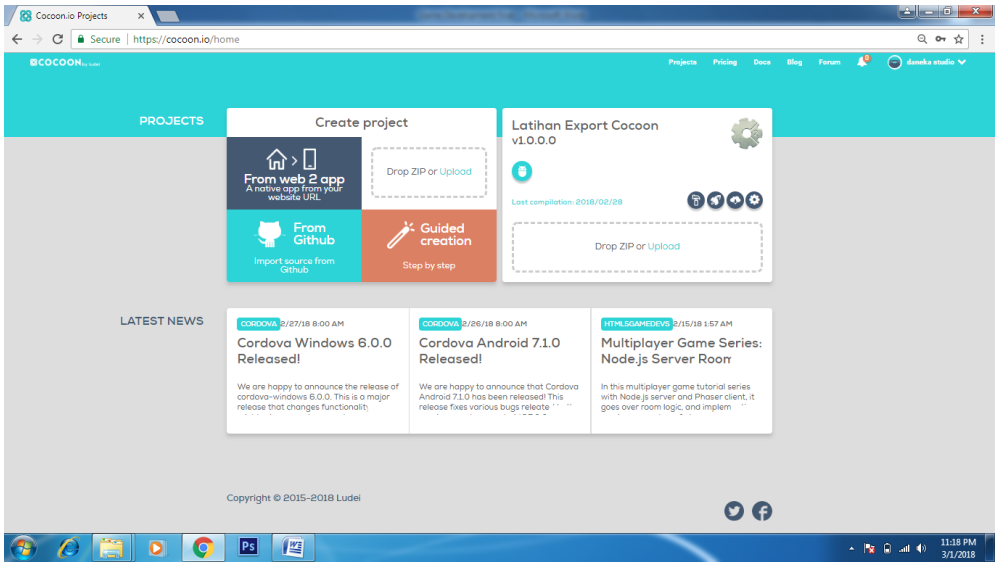
Gambar 11.34 Memilih keystore

2. Setelah menemukan file keystore copy file keystore ke dalam PC anda agar bisa digunakan saat memberikan keystore pada game kita di Cocoon Io.



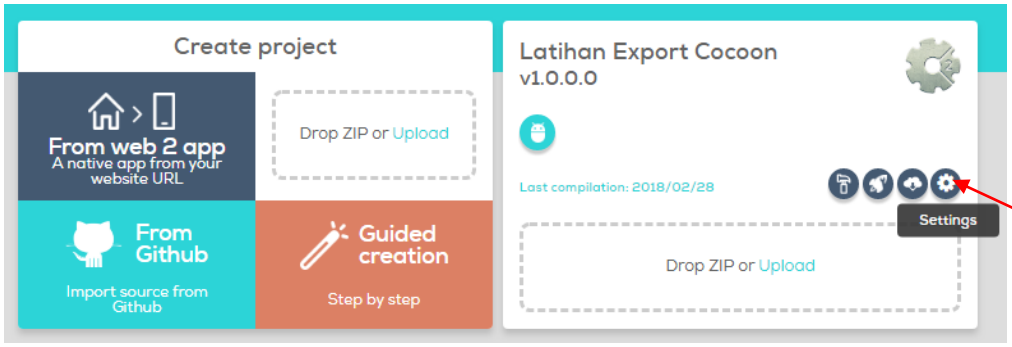
Gambar 11.35 Copy file keystore kedalam PC

3. Selanjutnya buka web https://cocoon.io/ dan masuk ke halaman utama untuk mengedit project yang sebelumnya kita buat pada praktikum 10.2.



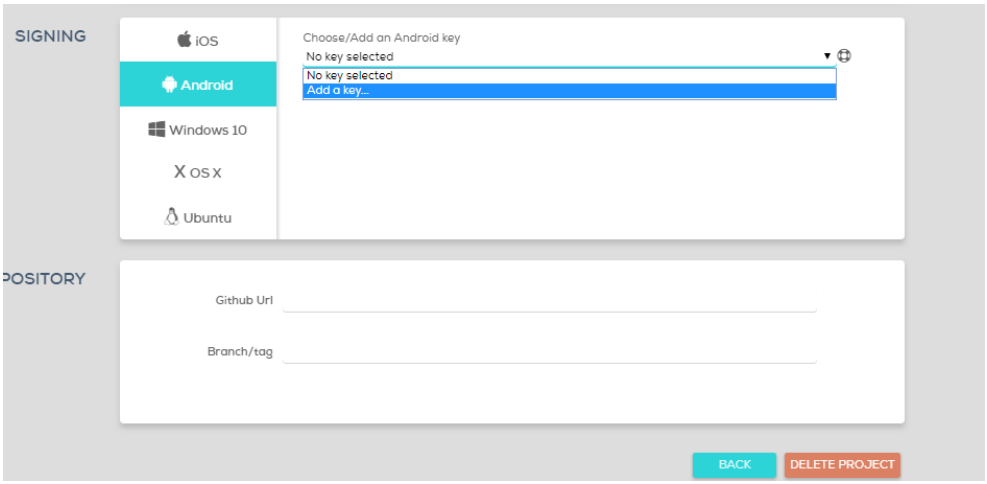
Gambar 11.36 Halaman utama Cocoon Io

4. Selanjutnya klik tombol setting untuk pergi ke halaman setting.



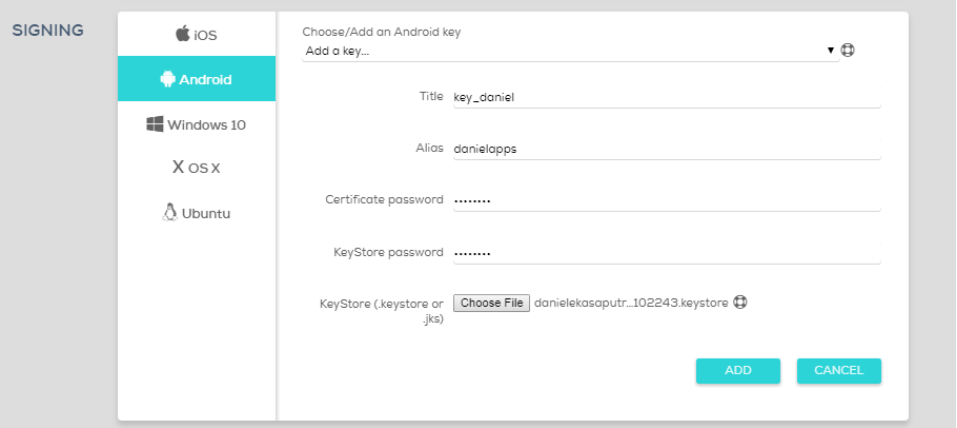
Gambar 11.37 Klik tombol setting

5. Selanjutnya pergi kebawah cari kolom Signing dan klik gambar android lalu klik Add a key untuk menambah keystore.

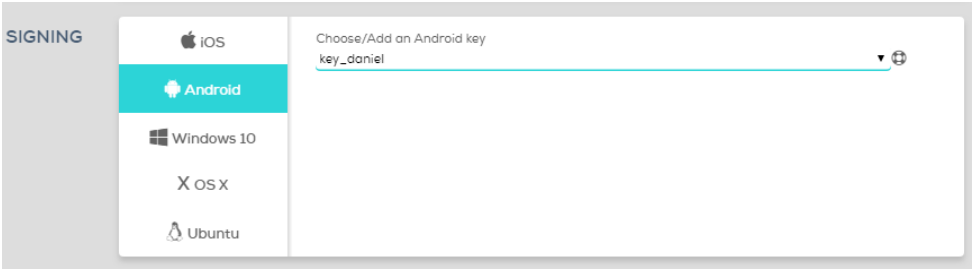


Gambar 11.38 Kolom Signing

6. Selanjutnya isi data dengan lengkap dan pilih Choose File untuk menammbah keystore yang sebelumnya sudah kita copy ke dalam PC. Setelah itu klik Add untuk menyimpan data isianya.

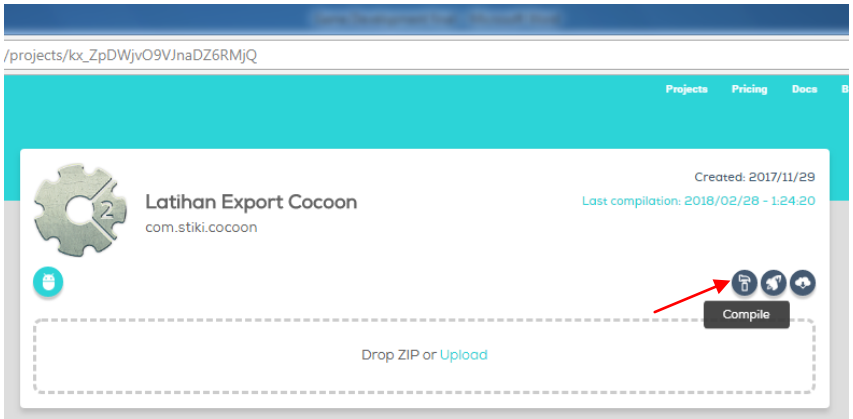


Gambar 11.39 Mengisi data pada kolom Signing



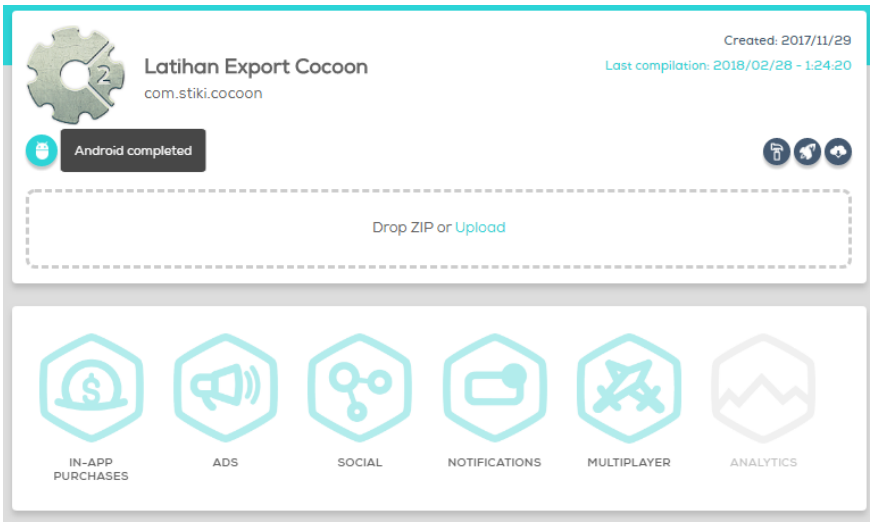
Gambar 11.40 Key yang sudah di add

7. Setelah key dibuat kita bisa lanjut ke tahap Compile. Klik tombol Compile untuk mengcompile ulang game yang kita buat.



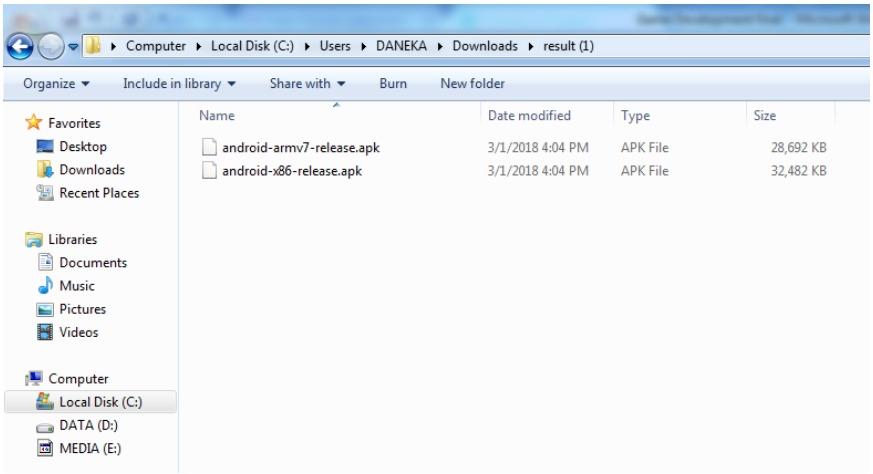
Gambar 11.41 Klik tombol Compile

8. Setelah itu tunggulah compile game beberapa saat sampai icon berwarna hijau dan berstatus completed.



Gambar 11.42 Compile Android Completed

9. Setelah itu klik icon android untuk mendownload hasil compile game Cocoon Io.



Gambar 11.43 Hasil download compile game Cocoon Io

Setelah game selesai download terlihat status file apk kita sudah release dan sudah bisa untuk di upload ke Android Market.