Farseer Physics (Box2D) and Unity (Part #2)

posted by gabs | Mar 27th, 21:09

Now that we got the basics of how our Farseer Physics Engine port works, let's move to more advanced things. In the last part I covered the **World**, **Body** and **Basic Shape** components... so what about concave shapes and joints?

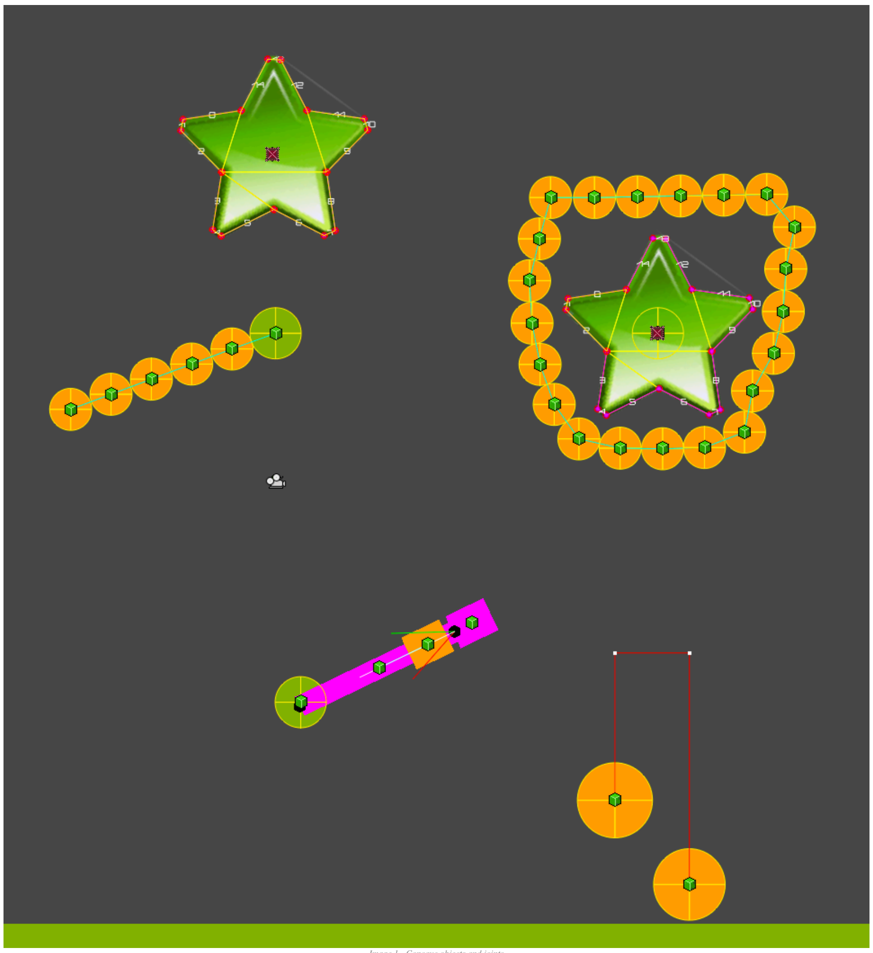


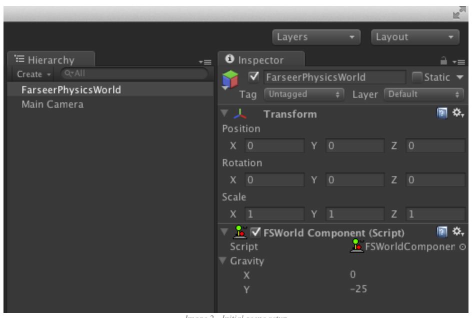
Image 1 - Concave objects and joints

Before starting this tutorial, download the FarseerUnity_CatsintheSky.unitypackage file then create a new project.

Concave Shapes

As of 99.9% of all physics engines out there, you can't have real concave shapes. If an engine is known to support concave shapes by default, it probably converts this shape into several smaller convex shapes. Doing this by hand can be very frustrating and time consuming but, luckily for us, Farseer includes several concave to convex decomposers.

Setup a scene with a camera (iso, size 27) and the FSWorldComponent like on the image below.



FarseerPhysicsWorld
Main Camera

FarseerPhysicsWorld Static
Tag Untagged Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
FSWorld Component (Script)
Script FSWorldComponer
Gravity
X 0
Y -25

Image 2 - Initial scene setup

Now, before we continue, we can add a simple test component to the FSWorldComponent's GameObject, you can find it as FSMouseTest (project tab) or like on Image 3.

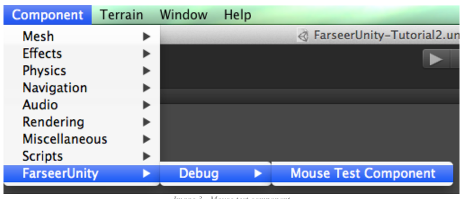
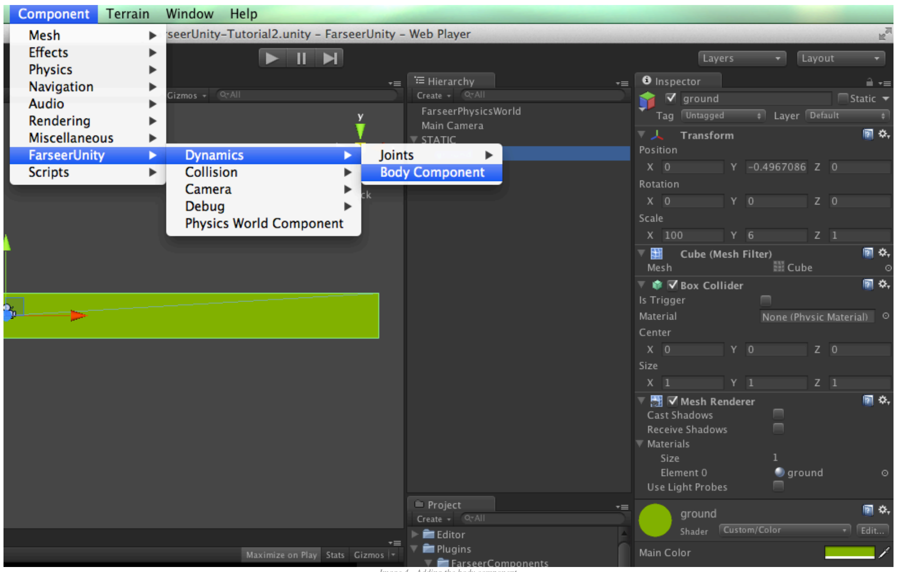


Image 3 - Mouse test component

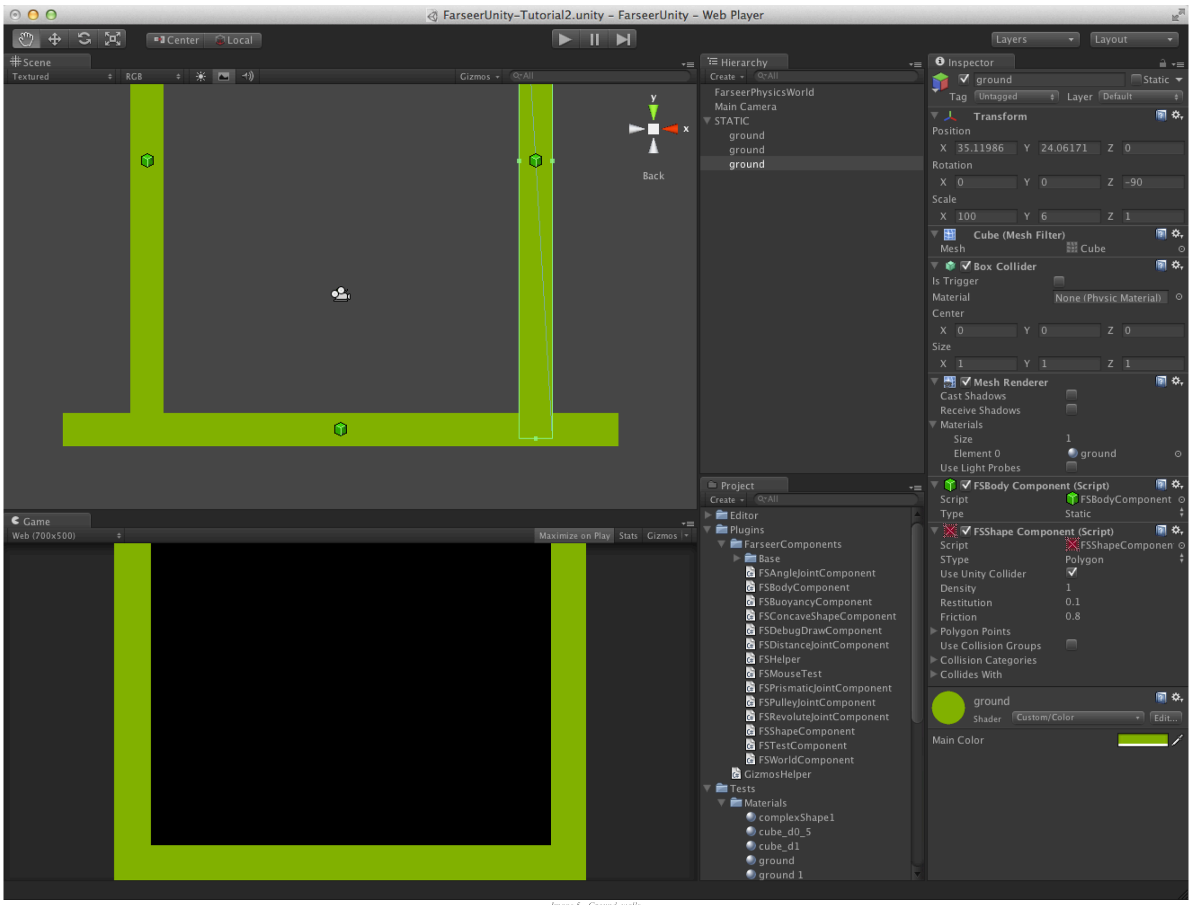
Now create a basic cube to set it as the ground. Once you got the dimensions (also materials, name, etc.) the way you want it, add the FSBodyComponent.



ground Static
Tag Untagged Layer Default
Transform
Position X 0 Y -0.4967086 Z 0
Rotation X 0 Y 0 Z 0
Scale X 100 Y 6 Z 1
Cube (Mesh Filter)
Mesh Cube
Box Collider
Is Trigger
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 1 Y 1 Z 1
Mesh Renderer
Cast Shadows
Receive Shadows
Materials
Size 1
Element 0 ground
Use Light Probes
ground
Shader Custom/Color Edit...
Main Color

Image 4 - Adding the body component

Add the regular FSShapeComponent at the same level as the FSBodyComponent remains. There's no need to put the shape component in a child object since we're using just one shape and we're borrowing Unity's collider to define the shape. After setting up the shape, I cloned the ground object to make the walls (Image 5).



Hierarchy
FarseerPhysicsWorld
Main Camera
STATIC
ground
ground
ground

Project
Editor
Plugins
FarseerComponents
Base
FSAngleJointComponent
FSBodyComponent
FSBuoyancyComponent
FSConcaveShapeComponent
FSDebugDrawComponent
FSDistanceJointComponent
FSHelper
FSMouseTest
FSPrismaticJointComponent
FSPulleyJointComponent
FSRevoluteJointComponent
FSShapeComponent
FSTestComponent
FSWorldComponent
GizmosHelper
Tests
Materials
complexShape1
cube_d0_5
cube_d1
ground
ground 1

Inspector
ground — Static
Tag Untagged — Layer Default
Transform
Position X 35.11986 Y 24.06171 Z 0
Rotation X 0 Y 0 Z -90
Scale X 100 Y 6 Z 1
Cube (Mesh Filter)
Mesh Cube
Box Collider
Is Trigger
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 1 Y 1 Z 1
Mesh Renderer
Cast Shadows
Receive Shadows
Materials
Size 1
Element 0 ground
Use Light Probes
FSBody Component (Script)
Script FSBodyComponent
Type Static
FSShape Component (Script)
Script FSShapeComponen
SType Polygon
Use Unity Collider ✓
Density 1
Restitution 0.1
Friction 0.8
Polygon Points
Use Collision Groups
Collision Categories
Collides With
ground
Shader Custom/Color
Main Color

Image 5 - Ground, walls

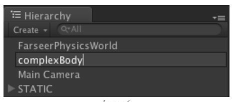
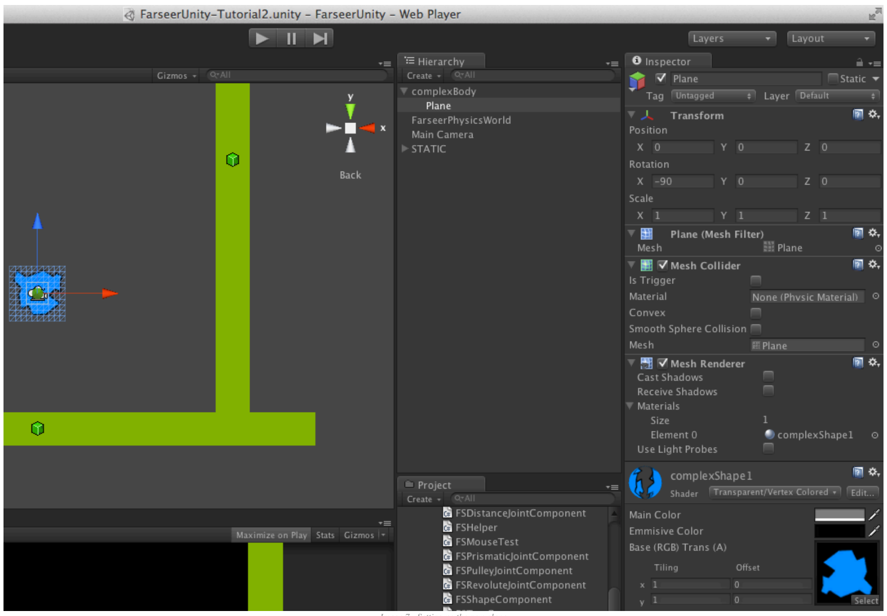

Hierarchy
FarseerPhysicsWorld
complexBody
Main Camera
STATIC

Image 6

Add a new GameObject because now we will setup a concave shape. Actually it is a component that easily converts an outline into convex polygons. I named mine as **complexBody** (Image 6).

Add a new plane (to use it as the graphics for our complex concave shape). Make it a child of the "complexBody" GameObject. Rotate it in the X axis for about -90 degrees. Add the complexShape1 material to it (Image 7).



Hierarchy
complexBody
Plane
FarseerPhysicsWorld
Main Camera
STATIC

Project
FSDistanceJointComponent
FSHelper
FSMouseTest
FSPrismaticJointComponent
FSPulleyJointComponent
FSRevoluteJointComponent
FSShapeComponent

Inspector
Plane — Static
Tag Untagged — Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X -90 Y 0 Z 0
Scale X 1 Y 1 Z 1
Plane (Mesh Filter)
Mesh Plane
Mesh Collider
Is Trigger
Material None (Physic Material)
Convex
Smooth Sphere Collision
Mesh Plane
Mesh Renderer
Cast Shadows
Receive Shadows
Materials
Size 1
Element 0 complexShape1
Use Light Probes
complexShape1
Shader Transparent/Vertex Colored
Main Color
Emissive Color
Base (RGB) Trans (A)
Tiling x 1 y 1
Offset x 0 y 0

Image 7 - Setting up the new plane

Add a new GameObject and make it a child of "complexBody" again. This will be the container of the component that handles the shape conversion (Image 8).

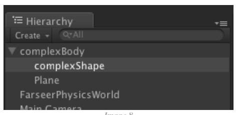


Image 8

Now, instead of adding the usual FSShapeComponent, we will add the FSConvexShapeComponent (Image 9).

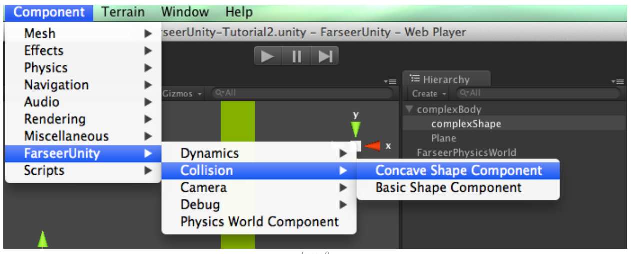


Image 9

Add a new GameObject, make it a child of "complexShape". I named mine as "p00". Setup its position (Image 10).

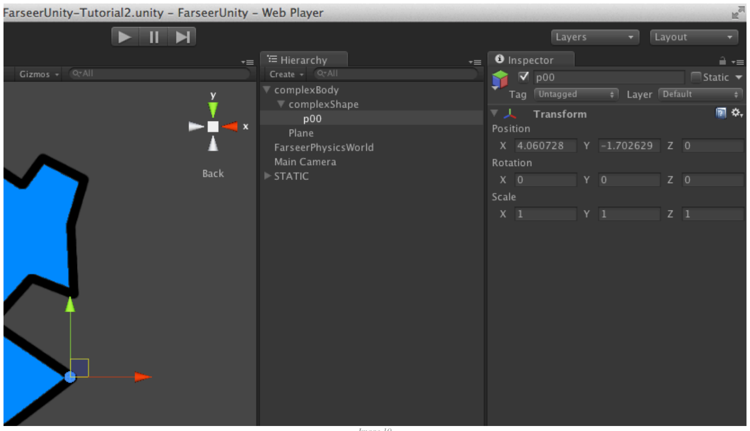


Image 10

Now start cloning the points and positioning them in a counter-clockwise order (Image 11, 12).

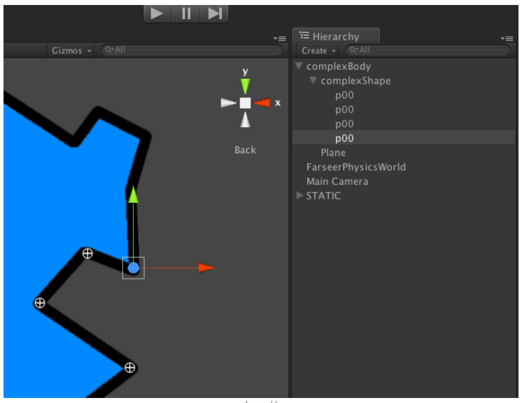
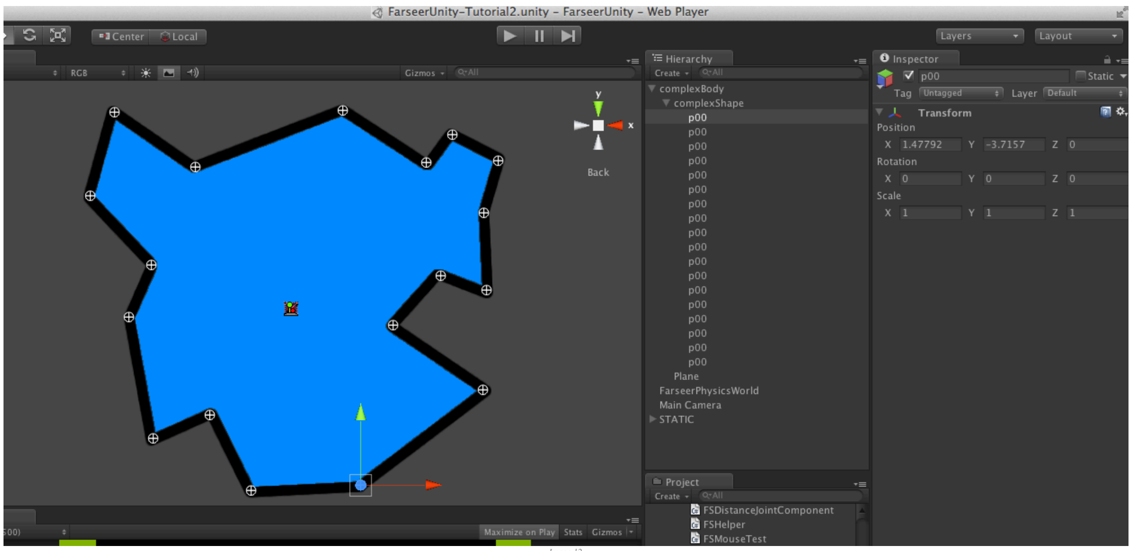


Image 11


FarseerUnity-Tutorial2.unity - FarseerUnity - Web Player

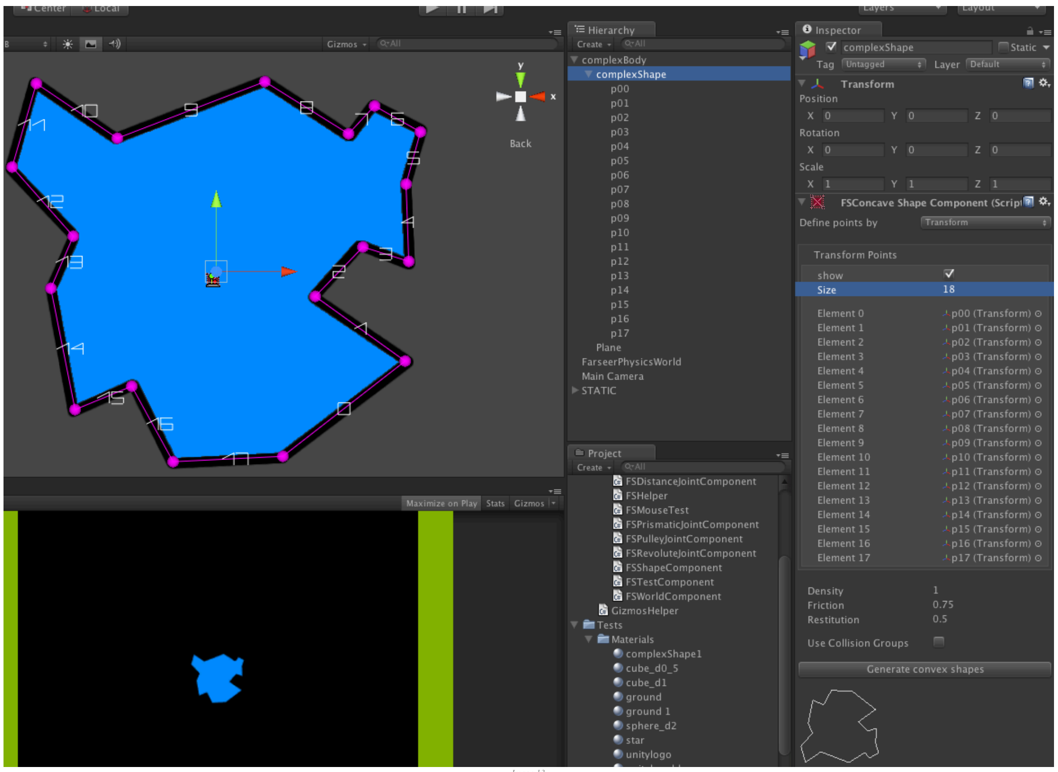
Hierarchy
complexBody
complexShape
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
p00
Plane
FarseerPhysicsWorld
Main Camera
STATIC

Inspector
p00 Static
Tag Untagged Layer Default
Transform
Position X 1.47792 Y -3.7157 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Project
FSDistanceJointComponent
FSHelper
FSMouseTest

Image 12

Now rename the remaining GameObjects and add them to the "Transform Points" array (Image 13). if the setup was done properly, you should see magenta points and line numbers.


Hierarchy
complexBody
complexShape
p00
p01
p02
p03
p04
p05
p06
p07
p08
p09
p10
p11
p12
p13
p14
p15
p16
p17
Plane
FarseerPhysicsWorld
Main Camera
STATIC

Inspector
complexShape Static
Tag Untagged Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
FSConcave Shape Component (Script)
Define points by Transform
Transform Points
show ✓
Size 18
Element 0 p00 (Transform)
Element 1 p01 (Transform)
Element 2 p02 (Transform)
Element 3 p03 (Transform)
Element 4 p04 (Transform)
Element 5 p05 (Transform)
Element 6 p06 (Transform)
Element 7 p07 (Transform)
Element 8 p08 (Transform)
Element 9 p09 (Transform)
Element 10 p10 (Transform)
Element 11 p11 (Transform)
Element 12 p12 (Transform)
Element 13 p13 (Transform)
Element 14 p14 (Transform)
Element 15 p15 (Transform)
Element 16 p16 (Transform)
Element 17 p17 (Transform)
Density 1
Friction 0.75
Restitution 0.5
Use Collision Groups
Generate convex shapes

Project
FSDistanceJointComponent
FSHelper
FSMouseTest
FSPrismaticJointComponent
FSPulleyJointComponent
FSRevoluteJointComponent
FSShapeComponent
FSTestComponent
FSWorldComponent
GizmosHelper
Tests
Materials
complexShape1
cube_d0_5
cube_d1
ground
ground 1
sphere_d2
star
unitylogo

Image 13

Click on the "Generate convex shapes" button so the real shapes will be generated. If you need to alter something later, you can press the button again to update the shapes (the old ones will be overwritten) (Image 14, 15).

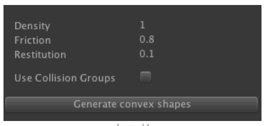

Density 1
Friction 0.8
Restitution 0.1
Use Collision Groups
Generate convex shapes

Image 14

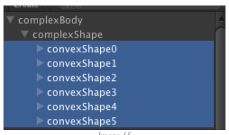

complexBody
complexShape
convexShape0
convexShape1
convexShape2
convexShape3
convexShape4
convexShape5

Image 15

The image below shows what the component generated. The outline was converted into 6 polygon shapes (Image 16).

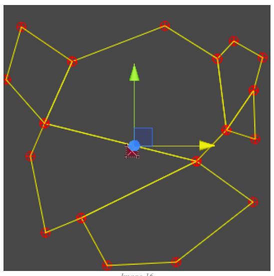


Image 16

The complex shape is done! Add a FSBodyComponent to "complexBody" if you didn't already. Then press play to test the collision of it. You can use the mouse to move objects around, since we added the mouse test component earlier.

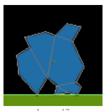
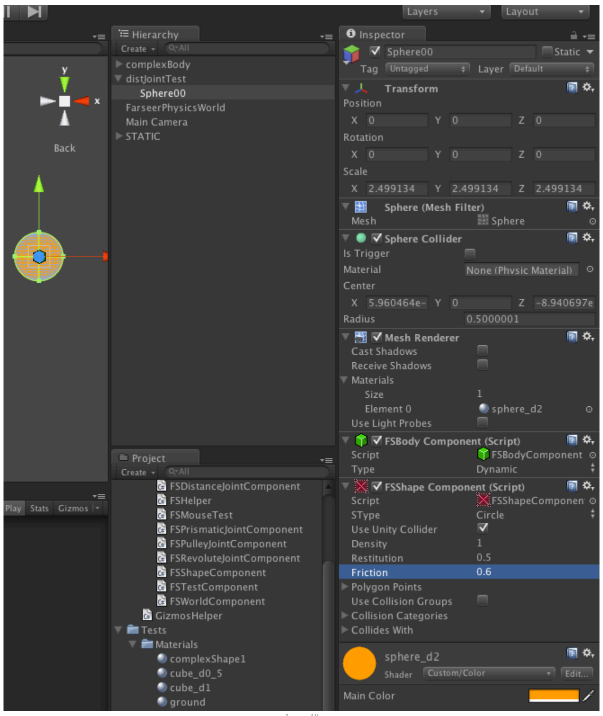


Image 17

Distance Joints

From the Box2D manual, *"one of the simplest joint is a distance joint which says that the distance between two points on two bodies must be constant"*. They are also the easiest ones to setup.

Start by making a sphere, attaching a FSBodyComponent then a FSShapeComponent. Set the shape type to circle then leave the "use unity collider" option checked (Image 18).



Sphere00 Static
Tag Untagged Layer Default

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 2.499134 Y 2.499134 Z 2.499134

Sphere (Mesh Filter)
Mesh Sphere

Sphere Collider
Is Trigger
Material None (Physic Material)
Center X 5.960464e- Y 0 Z -8.940697e
Radius 0.5000001

Mesh Renderer
Cast Shadows
Receive Shadows
Materials
Size 1
Element 0 sphere_d2
Use Light Probes

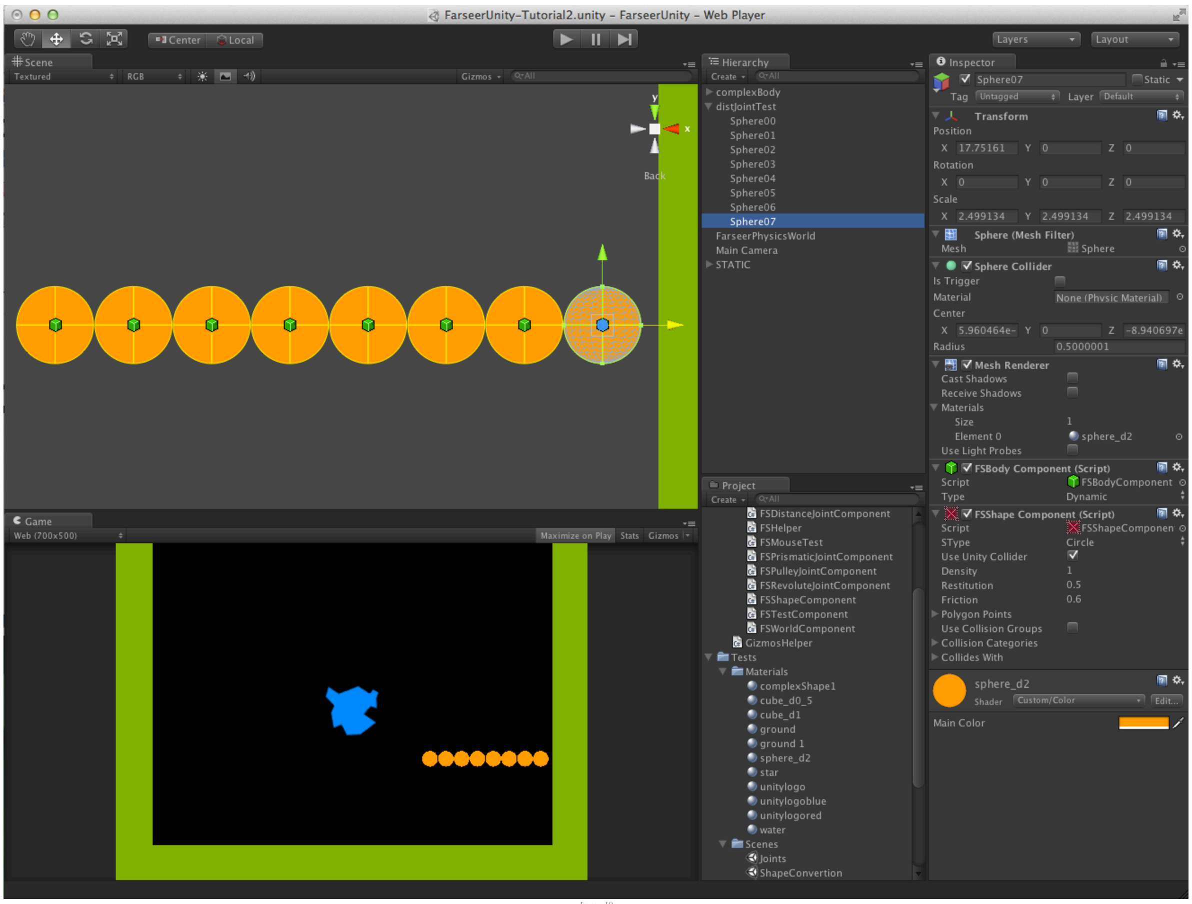
FSBody Component (Script)
Script FSBodyComponent
Type Dynamic

FSShape Component (Script)
Script FSShapeComponen
SType Circle
Use Unity Collider ✓
Density 1
Restitution 0.5
Friction 0.6
Polygon Points
Use Collision Groups
Collision Categories
Collides With

sphere_d2
Shader Custom/Color Edit...
Main Color

Image 18

Clone the spheres until you have about 8 of them. Rename their GameObjects so you won't get lost when setting the distance joints (Image 19).



Inspector
Sphere07 Static
Tag Untagged Layer Default

Transform
Position
X 17.75161 Y 0 Z 0
Rotation
X 0 Y 0 Z 0
Scale
X 2.499134 Y 2.499134 Z 2.499134

Sphere (Mesh Filter)
Mesh Sphere

Sphere Collider
Is Trigger
Material None (Physic Material)
Center
X 5.960464e- Y 0 Z -8.940697e
Radius 0.5000001

Mesh Renderer
Cast Shadows
Receive Shadows
Materials
Size 1
Element 0 sphere_d2
Use Light Probes

FSBody Component (Script)
Script FSBodyComponent
Type Dynamic

FSShape Component (Script)
Script FSShapeComponen
SType Circle
Use Unity Collider ✓
Density 1
Restitution 0.5
Friction 0.6
Polygon Points
Use Collision Groups
Collision Categories
Collides With

sphere_d2
Shader Custom/Color Edit...
Main Color

Image 19

Add a FSDistanceJointComponent. It doesn't matter where you add those joint components since they link bodies regardless of being a child of one of them, or not. I added one component per each sphere (except the last one) (Image 20).

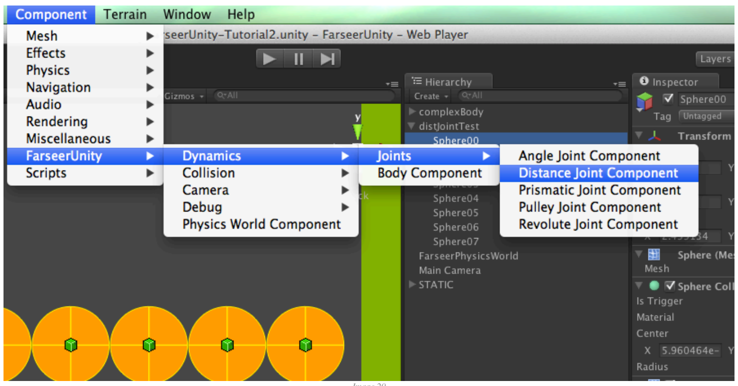


Image 20

Link BodyA and BodyB on each joint component. The pattern I used to link them was linking the object with the joint component as BodyA and the next body as BodyB (Image 21).

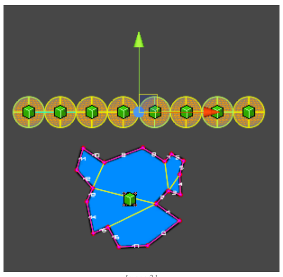
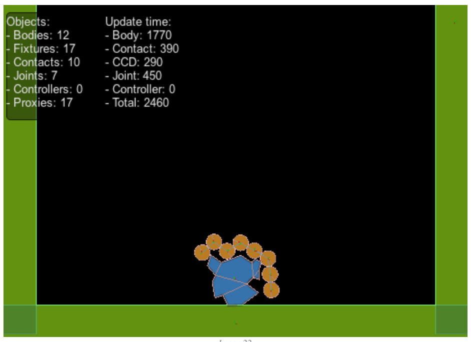


Image 21

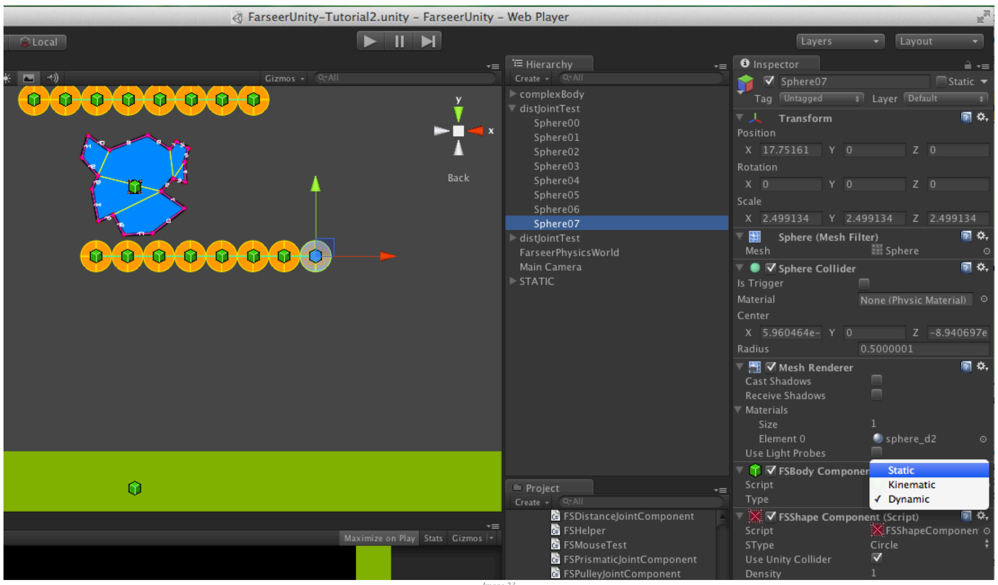
Once you linked everything, you can press play and test your creation (Image 22).



```
Objects:           Update time:
- Bodies: 12       - Body: 1770
- Fixtures: 17     - Contact: 390
- Contacts: 10     - CCD: 290
- Joints: 7        - Joint: 450
- Controllers: 0   - Controller: 0
- Proxies: 17      - Total: 2460
```

Image 22

After testing it, I cloned the connected bodies and turned one of the bodies as static to test this type of joint against static bodies too (Image 23, 24).



Hierarchy
- complexBody
- distJointTest
  - Sphere00
  - Sphere01
  - Sphere02
  - Sphere03
  - Sphere04
  - Sphere05
  - Sphere06
  - Sphere07
- distJointTest
- FarseerPhysicsWorld
- Main Camera
- STATIC

Project
- FSDistanceJointComponent
- FSHelper
- FSMouseTest
- FSPrismaticJointComponent
- FSPulleyJointComponent

Inspector
Sphere07 Static
Tag Untagged Layer Default

Transform
Position X 17.75161 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 2.499134 Y 2.499134 Z 2.499134

Sphere (Mesh Filter)
Mesh Sphere

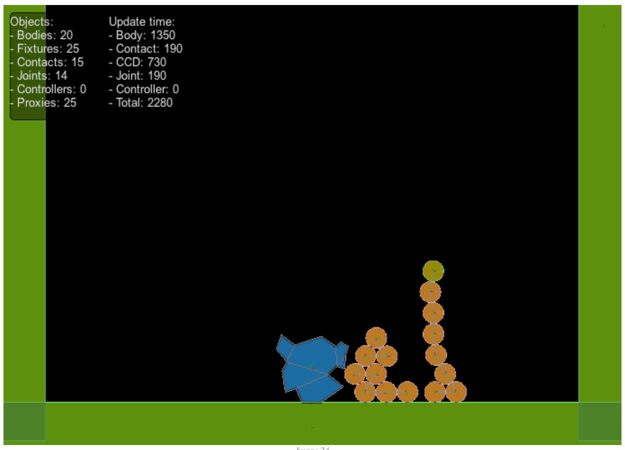
Sphere Collider
Is Trigger
Material None (Physic Material)
Center X 5.960464e- Y 0 Z -8.940697e
Radius 0.5000001

Mesh Renderer
Cast Shadows
Receive Shadows
Materials
Size 1
Element 0 sphere_d2
Use Light Probes

FSBody Component
Script
Type Static / Kinematic / ✓ Dynamic

FSShape Component (Script)
Script FSShapeComponen
SType Circle
Use Unity Collider ✓
Density 1

Image 23



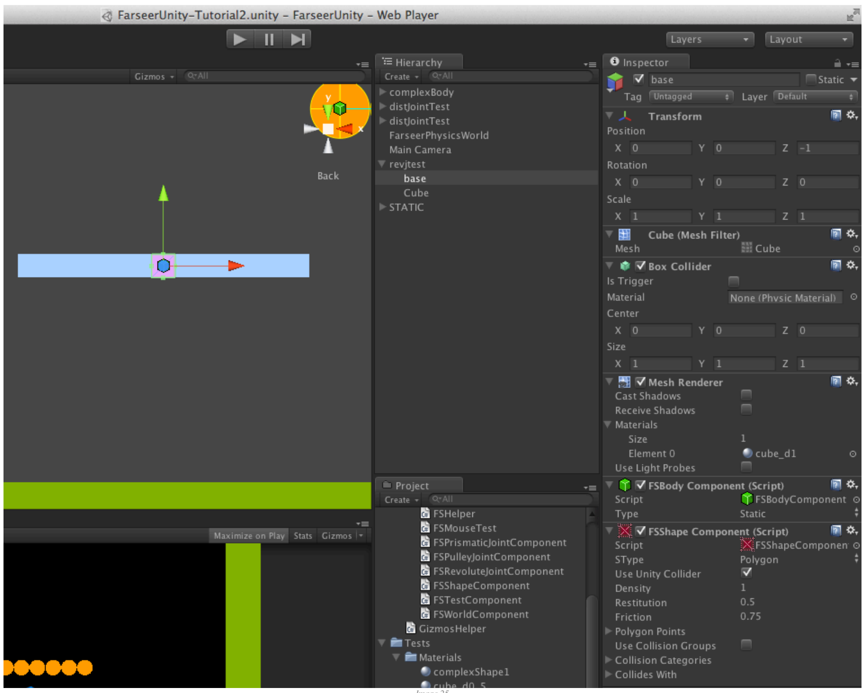
```
Objects:           Update time:
- Bodies: 20       - Body: 1350
- Fixtures: 25     - Contact: 190
- Contacts: 15     - CCD: 730
- Joints: 14       - Joint: 190
- Controllers: 0   - Controller: 0
- Proxies: 25      - Total: 2280
```

Image 24

Revolute Joints

"A revolute joint forces two bodies to share a common anchor point, often called a hinge point. The revolute joint has a single degree of freedom: the relative rotation of the two bodies. This is called the joint angle."

Add two new boxes and setup their bodies and their basic shape components. Make one of them smaller than the other and set the shape type as static (Image 25).


FarseerUnity-Tutorial2.unity - FarseerUnity - Web Player

Hierarchy: complexBody, distJointTest, distJointTest, FarseerPhysicsWorld, Main Camera, revjtest, base, Cube, STATIC

Inspector: base — Static
Tag Untagged, Layer Default

Transform
Position X 0 Y 0 Z -1
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Cube (Mesh Filter)
Mesh Cube

Box Collider
Is Trigger
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 1 Y 1 Z 1

Mesh Renderer
Cast Shadows
Receive Shadows
Materials
Size 1
Element 0 cube_d1
Use Light Probes

FSBody Component (Script)
Script FSBodyComponent
Type Static

FSShape Component (Script)
Script FSShapeComponen
SType Polygon
Use Unity Collider ✓
Density 1
Restitution 0.5
Friction 0.75
Polygon Points
Use Collision Groups
Collision Categories
Collides With

Project: FSHelper, FSMouseTest, FSPrismaticJointComponent, FSPulleyJointComponent, FSRevoluteJointComponent, FSShapeComponent, FSTestComponent, FSWorldComponent, GizmosHelper, Tests, Materials, complexShape1, cube_d0_5

Image 25

Add the FSRevoluteJointComponent to the base (smaller, static box) (Image 26).

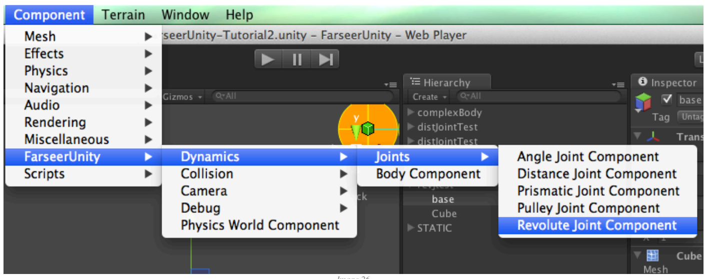


Image 26

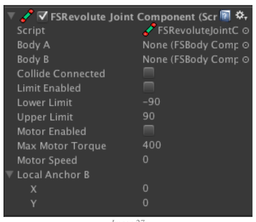

FSRevolute Joint Component (Scr
Script FSRevoluteJointC
Body A None (FSBody Comp
Body B None (FSBody Comp
Collide Connected
Limit Enabled
Lower Limit -90
Upper Limit 90
Motor Enabled
Max Motor Torque 400
Motor Speed 0
Local Anchor B
X 0
Y 0

Image 27

The FSRevoluteJointComponent has several options, unlike a distance joint. Both bodies will share the anchor point defined in Body B (Local Anchor B). Right now, it is set as [0,0], which means that both bodies will share the point at the center of Body B. The limit options can be used to limit the shared rotation of the connected bodies to a certain amount of degrees (this is commonly used when making ragdolls). The motor options can be used to apply an angular force at this shared point (commonly used when making wheels).

Add the smaller body (the base) to the Body A field, then add the other box to the Body B (Image 28).

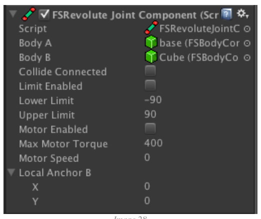
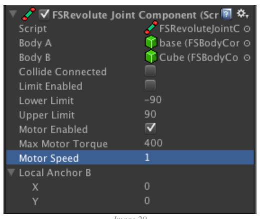

FSRevolute Joint Component (Scr
Script FSRevoluteJointC
Body A base (FSBodyCor
Body B Cube (FSBodyCo
Collide Connected
Limit Enabled
Lower Limit -90
Upper Limit 90
Motor Enabled
Max Motor Torque 400
Motor Speed 0
Local Anchor B
X 0
Y 0

Image 28

If you press play, you should see that if an object falls on top of it, the platform will rotate.

Now let's test the motor properties. Check the "Motor Enabled" field then set a speed other than zero (Image 29).



```
FSRevolute Joint Component (Scr
Script              FSRevoluteJointC
Body A              base (FSBodyCor
Body B              Cube (FSBodyCo
Collide Connected
Limit Enabled
Lower Limit         -90
Upper Limit         90
Motor Enabled       ✓
Max Motor Torque    400
Motor Speed         1
Local Anchor B
  X                 0
  Y                 0
```

Image 29

Now, to test the angle limit properties, clone both objects and move them away from the original objects, then move the platform to the right like on the image below.

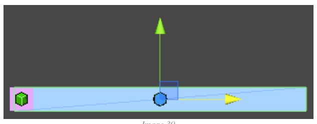
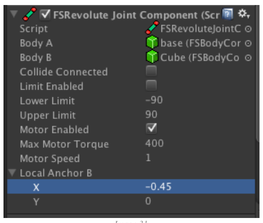


Image 30

Set the local anchor so the shared point will be at the left edge of Body B (Image 31, 32).



```
FSRevolute Joint Component (Scr
Script              FSRevoluteJointC
Body A              base (FSBodyCor
Body B              Cube (FSBodyCo
Collide Connected
Limit Enabled
Lower Limit         -90
Upper Limit         90
Motor Enabled       ✓
Max Motor Torque    400
Motor Speed         1
Local Anchor B
  X                 -0.45
  Y                 0
```

Image 31

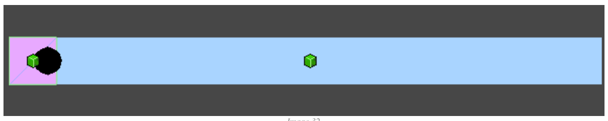


Image 32

Now check the "Limit Enabled" property. The white line is the current rotation of Body B. The rotation is limited by the green and red lines (Image 33, 34).

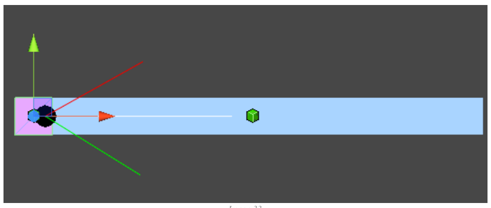
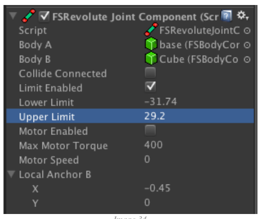


Image 33



```
FSRevolute Joint Component (Scr
Script              FSRevoluteJointC
Body A              base (FSBodyCor
Body B              Cube (FSBodyCo
Collide Connected
Limit Enabled       ✓
Lower Limit         -31.74
Upper Limit         29.2
Motor Enabled
Max Motor Torque    400
Motor Speed         0
Local Anchor B
  X                 -0.45
  Y                 0
```

Image 34

Clone the big platform, scale it, then move it to the right edge of the original platform (Image 35). Add a FSRevoluteJointComponent to it. The Body A will be the big platform and Body B will be this new object (Image 36).

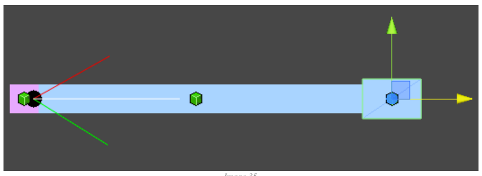


Image 35

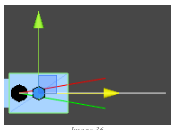


Image 36

Prismatic Joints

"A prismatic joint allows for relative translation of two bodies along a specified axis. A prismatic joint prevents relative rotation. Therefore, a prismatic joint has a single degree of freedom."

To test a prismatic joint: Clone this box again, change the color/material, remove the **revolute joint component** from it then move it like in the image below.

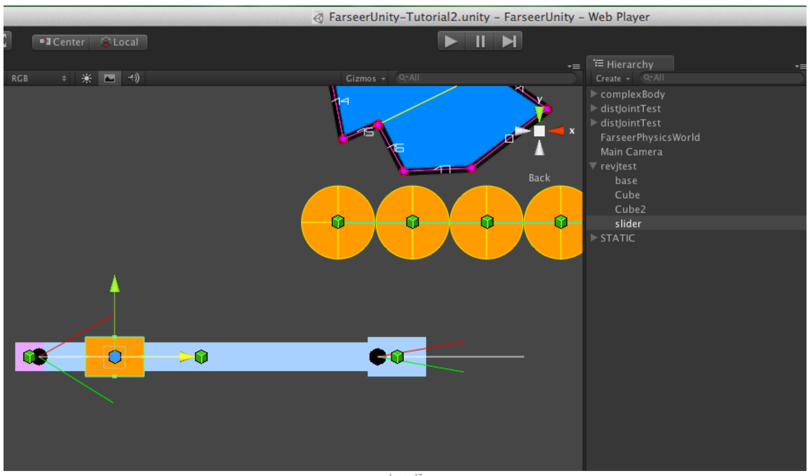


Image 37

Add a FSPrismaticJointComponent to it (Image 38).

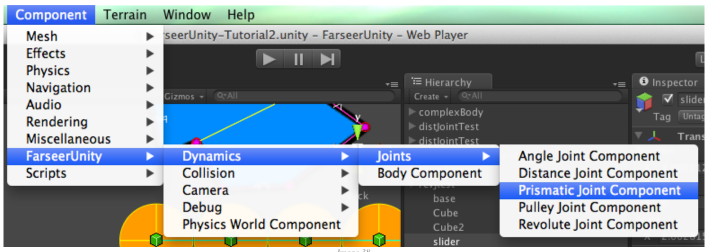
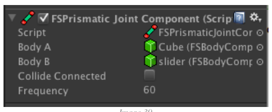


Image 38

The Body A is the big platform. I renamed the box to "slider". It's the Body B (Image 39).



FSPrismatic Joint Component (Scrip

Script	FSPrismaticJointCor
Body A	Cube (FSBodyComp
Body B	slider (FSBodyComp
Collide Connected	
Frequency	60

Image 39

Now test it again. If everything is correct, you should see something like in the picture below:

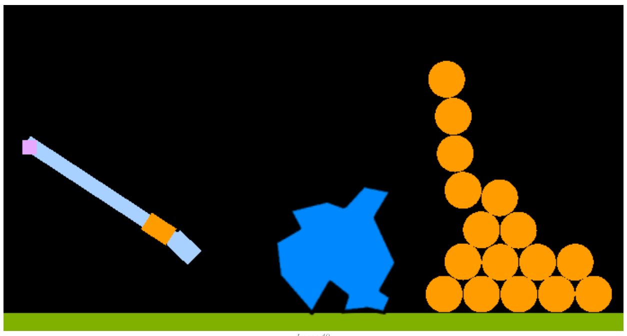


Image 40