CHAPTER 4 :

BODIES & JOINTS

Accessing Body /Joint in Script

Body or joint already existing in simulation can be accessed by calling getBody() and getJoint() functions of `'world'` object.

For example:

```
var d=World.getBody("Disc");
//stores body of name 'Disc' in variable d
var jt=World.getJoint("joint1");
//stores joint of name 'joint1' in variable jt
```

Creating Body / Joints

1. By Defining New Objects

New Body or joint can be created by calling `world.addXXX()` and `world.addXXXJoint()` functions.

The following code creates a rectangle and a disc connected by a spring of force constant 40 N/m.

```
var d = World.addDisc(0.5);
var r = World.addRectangle(1, 1);
r.setPosition(2, 0);
var j = World.addSpringJoint(d, r, null, null, 30, 0);
```

2. From Existing Objects

Clone of body can be created and added to world by calling function `createCopy(body)` in world object.

```
var d=World.getBody("Disc");
var d1=World.createCopy(d);
//creates copy of Disc and adds to world
```

Clones of body can also be created by calling `copy()` from body object, but note that this function creates copy of object and returns it without adding it to world.

This returned copy can later be added to world by using `world.addBody();`

```
var d=World.getBody("Disc");
var d1=d.copy();
World.addBody(d1);
```

Changing Appearance

There exist few functions in body object to change body's color, image and text.

```
var d=World.getBody("Disc");
d.setFillColor(“white”);
d.setOutlineColor(new Color("red"))
d.setBrush("image1");
//make sure animation with name "image1" already exists
d.setText("custom text");
var j=World.getJoint("joint");
j.setColor(new Color("green"));
```

Setting Text on Body

Body can act like a text display box by setting its text property using setText(text) method on body object text can be simple text or may contain shortcodes for multiple styles defined by comma separated attribute value pairs inside square brackets.

- ### Allowed attributes

  font = Name of currently loaded fonts

  x = x coordinate of text position in body local coordinates

  y = y coordinate of text position in body local coordinates

  color = CSS color string like "red" or "rgb(255,0,0)" etc

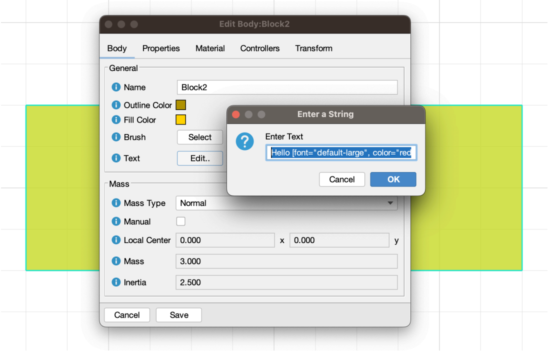
  xalign = 0, 1 or 2 for left, center and right respectively

  yalign = 0, 1 or 2 for bottom, up and down respectively

  For example,

  Following value of text set on rectangular body will result as in screenshot below.

```
Hello [font="default-large", color="red" alignX="left"
alignY="top"  x="-1",y="0.2"] Mr Anderson
[color="blue" x="0.03" y="0.1"] How r u?
```



Edit Body:Block2

General

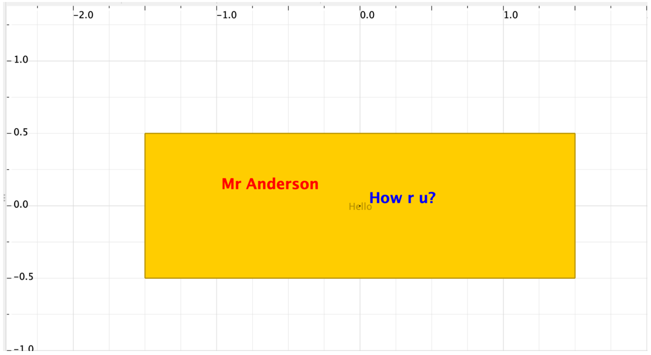
Name: Block2
Outline Color
Fill Color
Brush: Select
Text: Edit..

Enter a String

Enter Text: Hello [font="default-large", color="red

Mass

Mass Type: Normal
Manual: □
Local Center: 0.000 x 0.000 y
Mass: 3.000
Inertia: 2.500



Dynamics

All kinematic properties like position, velocity, rotation as well as dynamic properties like mass, inertia, force and torque can be accessed and set using script. for example the following code instantly brings body to origin and set is speed 3m/s in y direction (by applying suitable impulse)

```
var d = World.getBody("Disc");
d.setPosition(0, 0);
d.setVelocity(0, 0);
d.applyImpulse(new Vector2(0, 3).product(d.getMass()));
```

Actions

Each body has an action which is updated each frame internally. Many types of actions are included with SimPhy. These can be instantiated, configured, and added to a body. When the action is complete, it will automatically be removed from it.

Actions are generally used to perform some task , often over time, giving nice effect using specific interpolation.

Actions are created using methods of `actions` object, for example following code moves body from position (0,0) to (3,0) in 3 seconds following `"bounce"` interpolation.

```
var d = World.getBody("Disc");
//don't let other forces interact with body
d.setMass(0);
//set initial position of body
d.setPosition(0, 0);
//create action object
var a = Actions.moveTo(3, 0, 4, "bounce");
//set action to body
d.setAction(a);
```

Try adding the above code in some button click event run simulation, then click the button to see bouncy animation of Disc (assuming it exists in simulation ! For details about action see chapter dedicated to actions.

For details about action see chapter dedicated to actions.

Method Summary for Bodies

Method and Description	Modifier and Type
`applyForce(double fx, double fy)`	`void`
`applyImpulse(double jx, double jy)` Applies Impulse(in Newton-sec) on center of body	`void`
`applyImpulse(double jx, double jy, double px, double py)` Applies Impulse(in Newton sec) at specific point of body	`void`
`copy()` Returns the Copy of this body, the copied body is same in look and size, but is placed at origin and has speed zero	`Body`
`copy(boolean applyTransform)` Returns the Copy of this body, the copied body is same in look and size	`Body`
`getAction()` Returns action associated to the body	`Action`
`getAngularVelocity()`	`double`

Returns angular velocity of the body in radians per second	
`getBrush()`	`Brush`
`getCharge()`	`double`
`getFillColor()` Returns the fill color.	`Color`
`getHeight()` Returns width of bounding box of non transformed body	`double`
`getInertia()` Get moment of inertia of body about its center of mass(return 0 of body is static)	`double`
`getMass()` Get mass of body (return 0 of body is static)	`double`
`getName()` Returns the name of the body.	`String`
`getOpacity()`	`int`
`getOutlineColor()` Returns the outline color.	`Color`
`getPosition()` Position of center of mass of body in world coordinates	`Vector2`
`getRotation()`	`double`

Returns rotation in radians CCW as positive	
`getText()` Returns text associated with body	`String`
`getUserData()` Returns custom user data associated with body	`Object`
`getVelocity()` Returns velocity of center of mass in world coordinates	`Vector2`
`getWidth()` Returns height of bounding box of non transformed body	`double`
`getzOrder()` Returns z-order of the body	`int`
`isFbdDrawn()` True, if free body diagram of body is to be drawn	`boolean`
`isRenderable()` Returns Sets visibility of body	`boolean`
`isSensor()` Returns true if body doesn't sense and process collision	`boolean`
`isTouchable()`	`boolean`
`rotate(double th)` Sets rotation about its center of mass.	`void`

scaleBy(double xScale, double yScale) Scales body in each direction	void
scaleTo(double xScale, double yScale) Scales body in each direction	void
setAction(Action action) Sets Action of body	void
setAngularVelocity(double w) Sets Angular Velocity of body	void
setAnimation(Brush brush)	void
setBrush(Brush brush)	void
setCharge(double charge)	void
setFDrawn(boolean fbdDrawn) Sets free body diagram of the body enabled	void
setFillColor(Color color) Sets Fill Color of the body (to disable filling shape pass null ar arguement)	void
setFillColor(float r, float g, float b, float a) Sets outline color of body	void
setFriction(double mu) Sets friction for body	void
setInertia(double I)	void

Set Moment of inertia of body about center of mass	
setMass(double m) Set mass of body	void
setName(String name) Sets the name of the body.	void
setOpacity(int opacity) Sets opacity of body	void
setOutlineColor(Color color) Sets Outline Color of the body (to disable rendering outline pass null ar arguement)	void
setOutlineColor(float r, float g, float b, float a) Sets outline color of body	void
setPosition(double x, double y) Sets position of body in world coordinates	void
setPosition(Vector2 v) Set position of body	void
setRenderable(boolean renderable) Sets visibility of body	void
setRestitution(double e) Set coefficient of restitution	void
setRotation(double th) Sets rotation about its center of mass.	void

setSensor(boolean sensor) Sets if body can sense and process collision	**void**
setSize(double width, double height) Sets size of body such that width and height in parameter become size of bounding box	**boolean**
setText(String s) Sets text associated with body	**void**
setTouchable(boolean touchable)	**void**
setUserData(Object userData) Sets custom user data for this body	**void**
setVelocity(double vx, double vy) Sets linear velocity of body in world coordinates	**void**
setzOrder(int zOrder) Sets z-order of the body	**void**
translate(double x, double y) Translates body in world	**void**
translateToOrigin() Moves body such that its center lies at world origin	**void**

Method Summary for Joints

Method and Description	Modifier and Type
getAnchor1 ()	abstract **Vector2**

Returns the anchor point on the first Body in world coordinates.	
getAnchor2 () Returns the anchor point on the second Body in world coordinates.	abstract **Vector2**
getColor ()	**Color**
getName () Returns the name of the joint.	**String**
getOpacity ()	**int**
getReactionForce (double invdt) Returns the force applied to the Body in order to satisfy the constraint in Newtons.	abstract **Vector2**
getReactionTorque (double invdt) Returns the torque applied to the Body in order to satisfy the constraint in newton-meters.	abstract **double**
getSize () Size of the joint	**float**
getUserData () Returns custom user data associated with joint	**Object**
isCollisionAllowed () Returns true if collision between the joined Body is allowed.	**boolean**
isRenderable ()	**boolean**
setCollisionAllowed (boolean flag)	**void**

Sets whether collision is allowed between the joined Bodys.	
setColor (Object color) Sets color with which joint should be rendered	void
setName (String name) Sets the name of the joint.	void
setOpacity (int opacity) Sets opacity of joint	void
setRenderable (boolean renderable)	void
setSize (float size) Size of the joint (size=1 means default size) size >1 scales up size while size<1 scales down the size (max size=5)	void
setUserData (Object userData) Sets custom user data for this joint	void