# TNT Collision Engine v1.05

*iOS & Android native code Plugin for Gideros Mobile SDK*

*part of TNT Engine Framework*



*TNT Engine is composed by*

*Particles Engine*

*Virtual Pad*

*Animator Studio*

*Collision Engine*

for more info and news please visit www.tntengine.com or

follow me on  Twitter @GregBUG.

*Dedicated to my wife "Cinzia" and my son "Mattia".*

# Introduction

TNT Collision Engine is a set of functions to detect collision between to sprites (or video regions) these can be points, boxes, circles or oriented boxes regions.

This regions can be uses as collision "mask" to detect when two objects collides.

TNT Collision engine is optimized for speed and easy usage so no physics behavior are used/returned/calculated; if you need them is better to use a physics engine (like Box2D that is integrated with Gideros SDK).

You can use TNT Collision for your *free or commercial projects without limits*, but if you use it please support me with a donation.

Donations are very important for me; if you want to see updates, bugfix of this library support my work with a donation! :)

Thanks for Your Support,

*Gianluca D'Angelo*

gregbug@gmail.com

# Installation

TNT Collision engine is a c++ native plugin so the installation differ from other libraries:
if you already know how to install native plugins on Gideros you can skip this part.

## Desktop Player Windows

Installation on Windows Desktop Player is very easy; simply follow this steps:

1. Close your desktop player if it's open.
2. Go on TNT Collision Engine folder and select "tntNativeCollision.dll" (under libs\winplayer folder)
3. Copy "tntNativeCollision.dll" under Gideros Plugins folder on my PC (windows 7 64 bit) for example is located in "C:\Program Files (x86)\Gideros\Plugins"
4. Ready to go !

## Desktop Player OSX

Same as Windows installation:

1. Close your desktop player if it's open.
2. Go on TNT Collision Engine folder and select "libtntcollision.dylib" (under libs\osx folder)
3. Copy "libtntcollision.dylib" under Gideros Plugins folder.
4. Ready to go !

## Gideros Android Player

To use TNT Collision Engine with the device player you need to recompile Gideros Player, but it's very simple:

1. Start Gideros Studio and create a project with name MyGiderosAndroidPlayer (File -> New Project)
2. Without adding any files, export it as an Eclipse project (File -> Export Project -> Andorid)
3. Go in your just created folder (MyGiderosAndroidPlayer ) and delete the directory assets.
4. Copy from TNT Collision folder *Libs/Android* "armeabi" and "armeabi-v7a" folders into MyGiderosAndroidPlayer/libs folder
5. Open java source file **MyGiderosAndroidPlayerActivity.java** located in *MyGiderosAndroidPlayer /src/com/giderosmobile/android* and add under the line

   ```
   System.loadLibrary("gideros");
   System.loadLibrary("tntcollision");
   ```

Now you can import in your eclipse and deploy as every standard project then install on your device.

## Gideros iOS Player

1. Decompress GiderosiOSPlayer.zip
2. Open giderosiOSPlayer.xcodeproj with xCode
3. Add under GiderosiOSPlayer/Plugins the file libtntcollision.a (from TNT Collision Engine folder Libs/iOS)
4. Add in xCode under "Build Setting" of GiderosiOSPlayer Linking->"Other Linker Flags" the flag "-all_load" (without quotes)

ready for test/compile !

If you get linker error (ex: library not found for -luasocket)
with xcode 5 you need to correct the path
under
Build Setting
-> Search Path
LIBRARY_SEARCH_PATHS
this is the automatic generated string (in my system)

\"$(SRCROOT)/GiderosiOSPlayer\"
\"$(SRCROOT)/GiderosiOSPlayer/Plugins\"
/Volumes/Archivio/gregbug/Desktop/GiderosiOSPlayer/GiderosiOSPlayer

and this is the corrected version...

/$"(SRCROOT)/GiderosiOSPlayer/"
/$"(SRCROOT)/GiderosiOSPlayer/Plugins/"
/Volumes/Archivio/gregbug/Desktop/GiderosiOSPlayer/GiderosiOSPlayer

basically you need to change all "\" with "/"

also check this: Problems with XCode paths (fix)

## Android Compiled project

Same process as compiling "Android Player" but with your real project instead of "MyGiderosAndroidPlayer". (do not delete Asset folder!)

## iOS Compiled project

1. On Gideros Studio export your project with "File->Export Project.>iOS"
2. Open you project "yourProjectName" with xCode
3. add under "yourProjectName"/Plugins the file libtntcollision.a (from TNT Collision Engine folder Libs/iOS)
4. Add in xCode under "Build Setting" of "yourProjectName" Linking->"Other Linker Flags" the flag "-all_load" (without quotes)

ready for test/compile !

# Collision Anchor Point

## function setCollisionAnchorPoint(x, y)

Set current collision anchor point and MUST
coincide with Gideros setAnchorPoint()

**PARAMETERS**:
x: (number) The x coordinate of anchor point. Usually between [0, 1]
y: (number) The y coordinate of anchor point. Usually between [0, 1]

## function getCollisionAnchorPoint()

**RETURNS**
current x and y coordinates of the collision anchor point.

# Point Collisions

## function pointToBox(pointX, pointY, boxX, boxY, boxW, boxH)

This function check if a point collide or is inside AN AXIS ALIGNED (not rotated) Box.
Box CAN'T be rotated, but can be any Width/Height Size and can have any Anchor Point.

**PARAMETERS**:
pointX, pointY : coords of point to test
boxX, boxY : x, y coords of box
boxW, boxH : width and height of box
**RETURNS**
TRUE if collision else FALSE

EXAMPLE/TEST PROJECT: pointToBox_Collision_StressTest

## function pointToCircle(pointX, pointY, circleX, circleY, circleRadius)

This function check if a point collides with circle. Circle MUST BE AXIS ALIGNED or can be rotated if Anchor Point is set to (.5, .5) (centered).
Circle Width/Height MUST be the same Size (so no oval collision)
Can handle any Anchor Point value

**PARAMETERS**:
pointX, pointY : X and Y point coords
circleX, circleY : X and Y Circle coords

circleRadius : circle Radius

**RETURNS**

TRUE if collision else FALSE

EXAMPLE/TEST PROJECT: pointToCircle_Collision_StressTest

## function pointToObox(pointX, pointY, boxX, boxY, boxW, boxH, boxAngle)

This function check for point to ORIENTED Box collision. Box CAN BE ANY ANGLE ROTATION.
can handle any anchor point value, box Width/Height can be any size

**PARAMETERS**:

pointX, pointY : X and Y point coords
boxX, boxY : X and Y box coords
boxW, boxH : box Width and height
boxAngle : angle (in degree) of rotated box

**RETURNS**

TRUE if collision else FALSE

EXAMPLE/TEST PROJECT: pointToObox_Collision_StressTest

# Circle Collisions

## function circleToCircle(circleAx, circleAy, circleAradius, circleBx, circleBy, circleBradius)

This function check for circle to circle collision. Circles MUST BE AXIS ALIGNED
or can be rotated if Anchor Point is set to (.5, .5) (centered).
Circles Width/Height MUST be the same Size (so no oval collision)
Can handle any Anchor Point value

**PARAMETERS**:

circleAx, circleAy : X and Y circle coords
circleAradius : circle A Radius
circleBx, circleBx : X and Y Circle coords
circleBradius : circle B Radius

**RETURNS**

TRUE if collision else FALSE

EXAMPLE/TEST PROJECT: CircleToCircle_Collision, StressTest2

# Box Collisions

## function boxToBox(boxAx, boxAy, boxAw, boxAw, boxBx, boxBy, boxBw, boxBh)

This function check if two boxes collides. Boxes MUST BE AXIS ALIGNED (NOT rotated).
They can be any Width/Height Size and can have any Anchor Point.

**PARAMETERS**:
- boxAx, boxAy : Box A x and Y coords
- boxAw, boxAw : Box A Width and Height
- boxBx, boxBy : Box B x and Y coords
- boxBw, boxBh : Box B Width and Height

**RETURNS**
TRUE if collision else FALSE

EXAMPLE/TEST PROJECT: BoxToBox_CollisionTes, StressTest_1

## function boxToCircle(boxX, boxY, boxW, boxH, circleX, circleY, circleRadius)

This function check for box to circle collision. Box and Circles MUST BE AXIS ALIGNED
(not rotated) or can be rotated if Anchor Point is set to (.5, .5) (centered).
Circle Width/Height MUST be the same Size (so no Box to oval collision)
Can handle any Anchor Point value

**PARAMETERS**:
- boxX, boxY : X and Y box coords
- boxW, boxH : box Width and height
- circleX, circleY : X and Y Circle coords
- circleRadius : circle Radius

**RETURNS**
TRUE if collision else FALSE

EXAMPLE/TEST PROJECT: BoxToCircleCollision

## function oBoxToObox(boxAx, boxAy, boxAw, boxAh, boxAangle, boxBx, boxBy, boxBw, boxBh, boxBangle)

This function check for oriented box to oriented Box collision. Boxes CAN BE ANY ANGLE ROTATION.
can handle any anchor point value.
box Width/Height can be any size

**PARAMETERS**:
- boxAx, boxAy : X and Y box A coords
- boxAw, boxAh : box A Width and height
- boxAangle : angle (in degree) of rotated box A

boxBx, boxBy : X and Y box B coords
boxBw, boxBh : box B Width and height
boxBangle : angle (in degree) of rotated box B

RETURNS
TRUE if collision else FALSE

EXAMPLE/TEST PROJECT: oBoxToObox_Collision, oBoxToObox_StressTest

# Math Helper Functions

## function getDirection(x1, y1, x2, y2)

This function return angle between 2 points.

PARAMETERS:
x1, y1 : point a
x2, y2 : point b

RETURNS
direction (angle in degree) from point a to point b

## function getDistance(x1, y1, x2, y2)

This function return distance between 2 points.

PARAMETERS:
x1, y1 : point a
x2, y2 : point b

RETURNS
distance from point a to point b

## function getAngleDifference(angleA, angleB)

This function return smallest distance from angleA to angleB

PARAMETERS:
angleA : first angle
angleB : second angle

RETURNS
smallest distance from angleA to angleB in degree

## function getXLenDir(length, direction)

This function calculate X component from lenght and direction.

PARAMETERS:
- length : vector lenght
- direction : direction angle (degree)

RETURNS
- x component

## function getYLenDir(length, direction)

This function calculate y component from lenght and direction.

PARAMETERS:
- length : vector lenght
- direction : direction angle (degree)

RETURNS
- y component

## function getXYLenDir(length, direction)

This function calculate x and y component from lenght and direction.

PARAMETERS:
- length : vector lenght
- direction : direction angle (degree)

RETURNS
- (return 2 values) x and y component

## function min(a, b)

This function calculates min value from a and b.

PARAMETERS:
- a : first value
- b : second value

RETURNS
- smallest value between a and b

## function max(a, b)

This function calculates max value from a and b.

PARAMETERS:
- a : first value
- b : second value

RETURNS
biggest value between a and b

## function lerp(startValue, endValue, percentual)

Calculate linear interpolation from startValue to endValue with percentual

PARAMETERS:
- startValue : start value (ex. 10)
- endValue : end value (ex. 20)
- percentual : percentual or time for linear interpolation (ex. 0.2)

RETURNS
linear interpolation step value

## function clamp(value, min, max)

This function limit (clamp) value between min to max.

PARAMETERS:
- value : value to clamp
- min : min value possible
- max : max value possible

RETURNS
clamped (limited) value.

## function circularClamp(x1, y1, x2, y2, radius)

This function limit (clamp) a point (x2, y2) inside a circle defined with centre in x1, y1 and radius.

PARAMETERS:
- x1, y2 : circular centre x, y coordinates
- x2, y2 : point to clamp
- radius : circle radius

RETURNS
3 values clamped "x, y and radius"
circular clamped (limited) value. (point can't go out user defined circle)

## Release History

27/10/2012 v1.00 First Public Release
20/02/2014 v1.01 Added support for x86 library (Intel based Android devices)
06/03/2014 v1.05 Added Math Helper Functions