Справка по CGGeometry

Справка по CGGeometry описывает структуры для геометрических примитивов (базовых элементов), функции для работы с ними и константы для упрощения жизни. К примитивам относятся такие структуры как точка - CGPoint, размер - CGSize, прямоугольник - CGRect, двухмерный вектор CGVector а также список краев прямоугольника - CGRectEdge.

CGPoint - структура которая содержит точку в двухмерной системе координат.

```
struct CGPoint {

  CGFloat x;
  CGFloat y;

};
typedef struct CGPoint CGPoint;
```

Поля структуры CGPoint:

x - значение координаты по оси X, тип данных CGFloat.
y - значение координаты по оси Y, тип данных CGFloat.

Функции для работы со структурой CGPoint:

CGPointMake - функция которая создает и возвращает точку по заданным параметрам.

```
CGPoint CGPointMake ( CGFloat x, CGFloat y );
```

Пример:

```
CGPoint point = CGPointMake (10, 15);
```
Создастся точка point *с координатами* x = 10, y = 15;

CGPointEqualToPoint - функция которая сравнивает две точки и в возвращает булевое значение *true* (истина) или *false* (ложь).

```
bool CGPointEqualToPoint ( CGPoint point1, CGPoint point2 );
```

Пример:

```
CGPoint pointOne = CGPointMake (10, 15);
CGPoint pointTwo = CGPointMake (15, 10);
bool anyAnswer = CGPointEqualToPoint ( pointOne, pointTwo );
```
anyAnswer будет равно *false* так как точки не равны.

Константы для работы со структурой CGPoint:

const CGPoint CGPointZero - константа начала координат (0, 0). Тоже самое что и CGPointMake (0, 0).

CGSize - структура которая содержит ширину и высоту.

```
struct CGSize {

   CGFloat width;
```

```
    CGFloat height;

};
typedef struct CGSize CGSize;
```

Поля структуры CGSize:

width - ширина, тип данных CGFloat.
height - высота, тип данных CGFloat.

Функции для работы со структурой CGSize:

CGSizeMake - функция которая создает и возвращает размер в виде структуры.

```
CGSize CGSizeMake ( CGFloat width, CGFloat height );
```

Пример:

```
CGSize size = CGSizeMake ( 10, 20 );
```
Создастся размер size с с шириной width = 10, *и высотой* height = 20;

CGSizeEqualToSize - функция которая сравнивает два размера и возвращает булевое значение равны они или нет.

```
bool CGSizeEqualToSize ( CGSize size1, CGSize size2 );
```

Пример:

```
CGSize sizeOne = CGSizeMake ( 10, 20 );
CGSize sizeTwo = CGSizeMake ( 10, 20 );
bool anyAnswer = CGSizeEqualToSize ( sizeOne, sizeTwo );
```
anyAnswer будет равен *true* так как размеры одинаковы.

Константы для работы со структурой CGSize:

const CGSize CGSizeZero - константа создающая размер нулевой величины, тоже самое что и CGSizeMake (0, 0).

CGRect - структура прямоугольника которая содержит в себе структуру точки верхнего левого угла CGPoint а также структуру размера CGSize в которой содержатся ширина и высота прямоугольника.

```
struct CGRect {

    CGPoint origin;
    CGSize size;

};
typedef struct CGRect CGRect;
```

Поля структуры CGRect:

origin - структура CGPoint которая содержит координаты левого верхнего угла прямоугольника.

size - структура CGSize которая содержит значения ширины и высоты прямоугольника.

Функции для работы со структурой CGRect:

CGRectMake - функция которая создает и возвращает структуру прямоугольника с заданными координатами и размерами.

CGRect CGRectMake (CGFloat x, CGFloat y, CGFloat width, CGFloat height);

Пример:

CGRect rect = CGRectMake (10, 20, 50, 40);
Будет создан прямоугольник rect с координатами x = 10, y = 20, width = 50, height = 40;

CGRectEqualToRect - функция которая сравнивает два прямоугольника и возвращает булевое значение *true* (если равны) или *false* (если не равны).

bool CGRectEqualToRect (CGRect rect1, CGRect rect2);

Пример:

CGRect rect1 = CGRectMake (10, 20, 50, 40);
CGRect rect2 = CGRectMake (15, 25, 50, 40);
bool anyAnswer = CGRectEqualToRect (rect1, rect2);
anyAnswer будет равен *false* так как прямоугольники не равны.

CGRectContainsPoint - функция определяет находится ли нужная нам точка внутри прямоугольника и возвращает булевое значение *true* (если находится) или *false* (если нет).

bool CGRectContainsPoint (CGRect rect, CGPoint point);

Пример:

CGPoint point = CGPointMake (10, 15);
CGRect rect = CGRectMake (0, 0, 50, 40);
bool anyAnswer = CGRectContainsPoint (rect, point);
anyAnswer будет равен *true* так как точка находится в прямоугольнике

CGRectContainsRect - функция принимает два прямоугольника и определяет нахождение второго прямоугольника в первом, если находится то возвращается булевое значение *true* или *false* (если нет).

bool CGRectContainsRect (CGRect rect1, CGRect rect2);

Пример:

CGRect rect1 = CGRectMake (10, 10, 100, 100);
CGRect rect2 = CGRectMake (20, 20, 60, 60);
bool anyAnswer = CGRectContainsRect (rect1, rect2);
anyAnswer будет равен *true* так как rect2 находится внутри прямоугольника rect1.

CGRectIntersectsRect - функция принимает два прямоугольника, определяет пересекаются ли они (или один находится в другом) и возвращает булевое значение *true* (если пересекаются) или *false* (если нет).

bool CGRectIntersectsRect (CGRect rect1, CGRect rect2);

Пример:

```
CGRect rect1 = CGRectMake ( 10, 10, 100, 100 );
CGRect rect2 = CGRectMake ( 60, 60, 120, 120 );
bool anyAnswer = CGRectIntersectsRect ( rect1, rect2 );
```

anyAnswer будет равен *true* так как rect2 пересекается с прямоугольником rect1.

CGRectIntersection - функция принимает два прямоугольника и если они пересекаются то возвращает прямоугольник пересечения этих двух прямоугольников.

CGRect CGRectIntersection (CGRect r1, CGRect r2);

Пример:

```
CGRect rectOne = CGRectMake(50, 50, 100, 100);
CGRect rectTwo = CGRectMake(80, 80, 160, 160);
CGRect rectNew = CGRectIntersection(rectOne, rectTwo);
NSLog(@"rectNew: %@«,NSStringFromCGRect(rectNew));
```

Консоль:

```
2013-12-20 09:09:44.905 CGGeometryTest[6858:70b] rectNew: {{80, 80}, {70, 70}}
```

CGRectDivide - функция разделяет прямоугольник на два дополнительных.

```
void CGRectDivide ( CGRect rect, CGRect *slice, CGRect *remainder, CGFloat amount,
                    CGRectEdge edge );
```

Передаваемые параметры:

rect - исходный прямоугольник который будет разделен,
*slice - указатель на прямоугольник который мы отделим от исходного (отрежем ломтик)
*remainder - указатель на прямоугольник который является остатком после того как разделили (то что осталось от прямоугольника после того как отрезали ломтик),
amount - расстояние от края (edge) на которое надо поделить прямоугольник,
edge - край прямоугольника от которого отсчитываем расстояние для разделения (отрезки ломтика).

Пример:

```
CGRect rect1 = CGRectMake (0, 0, 100, 100);
CGRect slice;
CGRect remainder;
CGRectDivide(rect1, &slice, &remainder, 30, CGRectMinXEdge);

NSLog(@"rect1: %@", NSStringFromCGRect(rect1));
NSLog(@"slice: %@", NSStringFromCGRect(slice));
NSLog(@"remainder: %@", NSStringFromCGRect(remainder));
NSLog(@"CGRectMinXEdge: %d", CGRectMinXEdge);
```

Консоль:

```
2013-12-18 09:14:32.791 CGGeometryTest[4841:70b] rect1: {{0, 0}, {100, 100}}
2013-12-18 09:14:32.791 CGGeometryTest[4841:70b] slice: {{0, 0}, {30, 100}}
```

2013-12-18 09:14:32.792 CGGeometryTest[4841:70b] remainder: {{30, 0}, {70, 100}}
2013-12-18 09:14:32.792 CGGeometryTest[4841:70b] CGRectMinXEdge: 0

Функция разделила прямоугольник rect1: {{0, 0}, {100, 100}} на slice: {{0, 0}, {30, 100}}
и remainder: {{30, 0}, {70, 100}} , отсчет расстояния проводился от края имеющий минимальную координату по оси X (то есть от левого края).

CGRectInset - функция возвращает увеличенный или уменьшенный прямоугольник с той же центральной точкой что и прямоугольник переданный в параметрах.

CGRect CGRectInset (CGRect rect, CGFloat dx, CGFloat dy);

Передаваемые параметры:

rect - исходный прямоугольник,

dx - координата по оси X на сколько надо уменьшить или увеличить прямоугольник (отрицательное число увеличивает а положительное уменьшает прямоугольник),

dy - координата по оси Y на сколько надо уменьшить или увеличить прямоугольник (отрицательное число увеличивает а положительное уменьшает прямоугольник)

Пример:

```
CGRect rectOne = CGRectMake(50, 50, 100, 100);
CGRect rectTwo = CGRectInset(rectOne, 10, 10);

NSLog(@"rectOne: %@",NSStringFromCGRect(rectOne));
NSLog(@"rectTwo: %@«,NSStringFromCGRect(rectTwo));
```

Консоль:

2013-12-20 08:39:01.358 CGGeometryTest[6253:70b] rectOne: {{50, 50}, {100, 100}}
2013-12-20 08:39:01.358 CGGeometryTest[6253:70b] rectTwo: {{60, 60}, {80, 80}}

функция уменьшила прямоугольник 10 десять точек с каждой стороны и центр остался тем же.

CGRectIntegral - функция принимает прямоугольник и конвертирует его параметры к ближайшим целым значениям, после чего возвращает его.

CGRect CGRectIntegral (CGRect rect);

Передаваемые параметры:

rect - исходный прямоугольник,

Пример:

```
CGRect rectOne = CGRectMake(50.34, 50.48, 100.34, 100.67);
CGRect rectOneModif = CGRectIntegral(rectOne);

NSLog(@"rectOne: %@",NSStringFromCGRect(rectOne));
NSLog(@"rectOneModif: %@«,NSStringFromCGRect(rectOneModif));
```

Консоль:

2013-12-20 08:52:26.134 CGGeometryTest[6790:70b] rectOne: {{50.34, 50.48}, {100.34, 100.67}}
2013-12-20 08:52:26.134 CGGeometryTest[6790:70b] rectOneModif: {{50, 50}, {101, 102}}

функция привела значения с дробной точкой к целым числам.

CGRectOffset - функция принимает прямоугольник и координаты по X и Y и возвращает прямоугольник смещенный по этим координатам.

CGRect CGRectOffset (CGRect rect, CGFloat dx, CGFloat dy);

Передаваемые параметры:

rect - исходный прямоугольник,
dx - смещение по оси X,
dy - смещение по оси Y,

Пример:

```
CGRect rectOrigin = CGRectMake( 50, 50, 100, 100 );
CGRect rectOffset = CGRectOffset( rectOrigin, -10, 10 );

NSLog(@"rectOrigin: %@",NSStringFromCGRect(rectOrigin));
NSLog(@"rectOffset: %@",NSStringFromCGRect(rectOffset));
```

Консоль:

```
2013-12-23 08:38:43.275 CGGeometryTest[4079:70b] rectOrigin: {{50, 50}, {100, 100}}
2013-12-23 08:38:43.275 CGGeometryTest[4079:70b] rectOffset: {{40, 60}, {100, 100}}
```

координаты смещения могут быть как положительными так и отрицательными.

CGRectStandardize - функция получает прямоугольник и возвращает его но с положительными параметрами размеров прямоугольника.

CGRect CGRectStandardize (CGRect rect);

Пример:

```
CGRect rectOrigin = CGRectMake( 100, 100, -60, -40 );
CGRect rectStandardize = CGRectStandardize(rectOrigin);

NSLog(@"rectOrigin: %@",NSStringFromCGRect(rectOrigin));
NSLog(@"rectStandardize: %@«,NSStringFromCGRect(rectStandardize));
```

Консоль:

```
2013-12-23 08:48:23.058 CGGeometryTest[4177:70b] rectOrigin: {{100, 100}, {-60, -40}}
2013-12-23 08:48:23.059 CGGeometryTest[4177:70b] rectStandardize: {{40, 60}, {60, 40}}
```

оригинальный прямоугольник имел отрицательные значения ширины и высоты но был возвращен уже с положительными параметрами и измененноц точкой левого верхнего угла прямоугольника.

CGRectUnion - функция принимает два прямоугольника и возвращает наименьший прямоугольник в котором могут находится два переданных прямоугольника.

CGRect CGRectUnion (CGRect r1, CGRect r2);

Пример:

```
CGRect r1 = CGRectMake( 10, 10, 50, 50 );
CGRect r2 = CGRectMake( 75, 75, 25, 25 );
CGRect rectUnion = CGRectUnion(r1, r2);

NSLog(@"rectUnion: %@«,NSStringFromCGRect(rectUnion));
```

Консоль:

```
2013-12-23 09:00:35.911 CGGeometryTest[4208:70b] rectUnion: {{10, 10}, {90, 90}}
```

функция вернула прямоугольник у которого точка левого верхнего угла равна {10, 10} а правого нижнего {100, 100}, в который точно входят два наших передаваемых прямоугольника.

CGRectGetMinX, CGRectGetMinY - функция принимает прямоугольник и возвращает минимальную координату по оси X или Y.

```
CGFloat CGRectGetMinX ( CGRect rect );
CGFloat CGRectGetMinY ( CGRect rect );
```

Пример:

```
CGRect rect = CGRectMake( 10, 20, 50, 50 );
CGFloat minX = CGRectGetMinX(rect);
CGFloat minY = CGRectGetMinY(rect);

NSLog(@"minX: %.1f",minX);
NSLog(@"minY: %.1f",minY);
```

Консоль:

```
2013-12-24 08:47:16.341 CGGeometryTest[4503:70b] minX: 10.0
2013-12-24 08:47:16.341 CGGeometryTest[4503:70b] minY: 20.0
```

CGRectGetMidX, CGRectGetMidY - функции принимают прямоугольник и возвращают его координату центра по оси X или Y.

```
CGFloat CGRectGetMidX ( CGRect rect );
CGFloat CGRectGetMidY ( CGRect rect );
```

Пример:

```
CGRect rect = CGRectMake( 0, 0, 60, 60 );
CGFloat midX = CGRectGetMidX(rect);
CGFloat midY = CGRectGetMidY(rect);

NSLog(@"midX: %.1f",midX);
NSLog(@"midY: %.1f»,midY);
```

Консоль:

2013-12-24 08:54:03.233 CGGeometryTest[4539:70b] midX: 30.0
2013-12-24 08:54:03.233 CGGeometryTest[4539:70b] midY: 30.0

CGRectGetMaxX, CGRectGetMaxY - функции принимают прямоугольник и возвращают самую большую координату по оси X или Y(то есть правую нижнюю точку).

```
CGFloat CGRectGetMaxX ( CGRect rect );
CGFloat CGRectGetMaxY ( CGRect rect );
```

Пример:

```
CGRect rect = CGRectMake( 0, 0, 60, 60 );
CGFloat maxX = CGRectGetMaxX(rect);
CGFloat maxY = CGRectGetMaxY(rect);

NSLog(@"maxX: %.1f",maxX);
NSLog(@"maxY: %.1f»,maxY);
```

Консоль:

```
2013-12-24 09:00:16.467 CGGeometryTest[4562:70b] maxX: 60.0
2013-12-24 09:00:16.467 CGGeometryTest[4562:70b] maxY: 60.0
```

CGRectGetHeight, CGRectGetWidth функции принимают прямоугольник и возвращают его ширину или высоту.

```
CGFloat CGRectGetWidth ( CGRect rect );
CGFloat CGRectGetHeight ( CGRect rect );
```

Пример:

```
CGRect rect = CGRectMake( 0, 0, 50, 75 );
CGFloat rectWidth = CGRectGetWidth(rect);
CGFloat rectHeight = CGRectGetHeight(rect);

NSLog(@"rectWidth: %.1f",rectWidth);
NSLog(@"rectHeight: %.1f»,rectHeight);
```

Консоль:

```
2013-12-24 09:05:40.712 CGGeometryTest[4587:70b] rectWidth: 50.0
2013-12-24 09:05:40.713 CGGeometryTest[4587:70b] rectHeight: 75.0
```

CGRectIsEmpty - функция принимает прямоугольник и возвращает булевое значение. Если прямоугольник имеет какие то значения то возвращается *false* (ложь) а если прямоугольник имеет все нулевые значения то возвращается *true* (истина).

```
bool CGRectIsEmpty ( CGRect rect );
```

Пример:

```
CGRect rectOne = CGRectMake( 0, 0, 25, 25 );
bool rectOneEmpty = CGRectIsEmpty(rectOne);

CGRect rectTwo = CGRectMake( 0, 0, 0, 0 );
bool rectTwoEmpty = CGRectIsEmpty(rectTwo);
```

```
CGRect rectThree;
bool rectThreeEmpty = CGRectIsEmpty(rectThree);

NSLog(@"rectOne: %d",rectOneEmpty);
NSLog(@"rectTwoEmpty: %d",rectTwoEmpty);
NSLog(@"rectThreeEmpty: %d»,rectThreeEmpty);
```

Консоль:

```
2013-12-24 09:19:52.861 CGGeometryTest[4707:70b] rectOne: 0
2013-12-24 09:19:52.862 CGGeometryTest[4707:70b] rectTwoEmpty: 1
2013-12-24 09:19:52.862 CGGeometryTest[4707:70b] rectThreeEmpty: 0
```

Если в функцию передать не инициализированный прямоугольник то будет считаться что у ниго есть какието значения отличные от нуля и вернет f*alse.*

CGRectIsNull - функция принимает прямоугольник и возвращает булевое значение, если он является нулевым прямоугольником то *true* иначе *false*. Нулевой прямоугольник можно получить используя функцию **CGRectIntersection.**

```
bool CGRectIsNull ( CGRect rect );
```

Пример:

```
CGRect rectOne = CGRectMake(50, 50, 100, 100);
CGRect rectTwo = CGRectMake(220, 220, 20, 20);
CGRect rectNew = CGRectIntersection(rectOne, rectTwo);
bool rectNull = CGRectIsNull(rectNew);

NSLog(@"rectNull: %d»,rectNull);
```

Консоль:

```
2013-12-25 08:37:28.640 CGGeometryTest[7057:70b] rectNull: 1
```

Так как прямоугольники не пересекаются то в rectNew был передан нулевой прямоугольник, после чего мы это уточнили при помощи функции CGRectIsNull.

CGRectIsInfinite - в функцию передается прямоугольник и проверяется является ли он бесконечным. Если он бесконечен то возвращается булевое значение *true* если нет то *false*. Бесконечным он может быть когда у него не определена ширина или высота, используется в Core Image framework.

```
bool CGRectIsInfinite ( CGRect rect );
```

Пример:

```
CGRect rectInfinite = CGRectInfinite;
bool anyAnswer = CGRectIsInfinite(rectInfinite);

NSLog(@"rectInfinite: %d»,anyAnswer);
```

Консоль:

```
2013-12-27 16:16:43.009 CGGeometryTest[2025:70b] rectInfinite: 1
```

Создать бесконечный прямоугольник можно при помощи константы **CGRectInfinite**, и как видим из консоли то возвращена вункцией истина.

Константы для работы со структурой CGRect:

const CGRect CGRectInfinite - константа являющаяся бесконечным прямоугольником.

const CGRect CGRectZero - константа являющаяся нулевым прямоугольником, тоже самое что и CGRectMake(0,0,0,0).

const CGRect CGRectNull - константа являющаяся пустым прямоугольником.

CGRectEdge - список перечисляющий стороны прямоугольника.

```
enum CGRectEdge {
   CGRectMinXEdge,
   CGRectMinYEdge,
   CGRectMaxXEdge,
   CGRectMaxYEdge
};
typedef enum CGRectEdge CGRectEdge;
```

Параметры:

CGRectMinXEdge - это левый край прямоугольника,
CGRectMinYEdge - это верхний край прямоугольника,
CGRectMaxXEdge - это правый край прямоугольника,
CGRectMaxYEdge - это нижний край прямоугольника.

Пример использования данного списка можно увидеть в функции **CGRectDivide**.

CGVector - структура содержащая двумерный вектор.

```
struct CGVector {
   CGFloat dx;
   CGFloat dy;
};
typedef struct CGVector CGVector;
```

Параметры:

dx - координата вектора по оси X,
dy - координата вектора по оси Y

CGVectorMake - создает и возвращает вектор с указанными параметрами.

CGSize CGVectorMake (CGFloat dx, CGFloat dy);