

# Cours n°7 du 11 octobre 2012

3MCI n1 2012-2013

## Views

Dessiner avec Core Graphics

# View

- Une **vue** est une sous-classe de la classe **UIView**
  - **UIButton**, **UILabel**, **UITextField** sont des sous-classes de **UIView**
- C'est un rectangle situé par ses coordonnées dans lequel
  - il est possible de dessiner
  - les événements sont pris en charge
- Hiérarchie entre les **vues**
  - une **vue** a une seule superview – `(UIView *)superView;`
  - une **vue** peut avoir zéro ou plusieurs subviews – `(NSArray *)subView;`

3

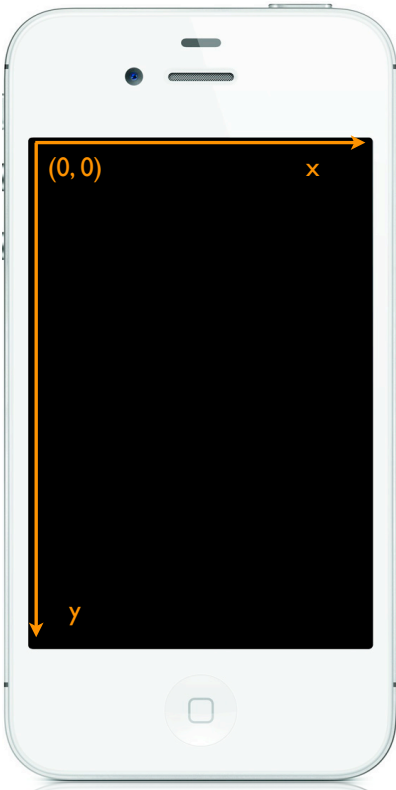
# Dessiner dans une **vue**

Librairie **Core Graphics**

- **CGFloat**: nombre à virgule flottante
- **CGPoint**: **C struct** qui définit un point situé par deux coordonnées **x** et **y** (**CGFloat**)
  - `CGPoint p = CGPointMake(77.5, 23.0);`
- **CGSize**: **C struct** qui définit une longueur et une largeur (**CGFloat**)
  - `CGSize s = CGSizeMake(150.0, 300.0);`
- **CRect**: **C struct** qui définit un rectangle avec un point d'origine (**CGPoint**) et une taille (**CGSize**)
  - `CRect unRect = CGRectMake(77.5, 23.0, 150.0, 300.0);`

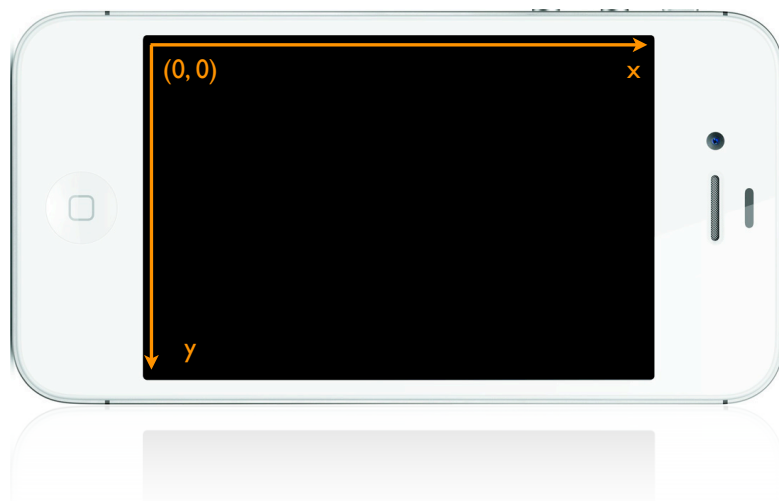
4

## Coordonnées



- ▶ **Origine dans le coin en haut à gauche**
- ▶ **Unités sont des points (pas des pixels)**
  - @property CGFloat contentScaleFactor;
- ▶ **3 propriétés:**
  - @property CGRect bounds;
  - @property CGPoint center;
  - @property CGRect frame;

5



## Rotation de l'appareil

- ▶ **L'origine des coordonnées est toujours au coin haut gauche**
  - La largeur devient plus grande que la hauteur
  - Plus de détails sur la gestion de l'autorotation la prochaine fois...

6

# Créer une vue

- **Dans Xcode:** à partir de la librairie d'objets à disposition (UIView générique)
- **Dans le code:** `[[... alloc] initWithFrame: ...];`

Exemple:

```
CGRect rectanglePourUnLabel = CGRectMake(50, 50, 100, 40);
UILabel *label = [[UILabel alloc] initWithFrame:rectanglePourUnLabel];
label.text = @"Bonjour!";
[self.view addSubview: label];
```

7

# Quand veut-on créer ses propres Views?

Et sa propre sous-classe de UIView

- **Lorsque l'on veut dessiner sur l'écran**
- **Lorsque l'on veut gérer certains événements tactiles particuliers**

8

# Comment est-ce qu'on dessine?

- `(void)drawRect(CGRect)unRect;`
- ▶ **Ne jamais appeler `drawRect` directement !**
  - ▶ C'est iOS qui s'occupe d'appeler cette méthode
- ▶ **Au contraire, faire savoir à iOS que notre interface a besoin d'être rafraîchie**
  - ▶ - `(void)setNeedsDisplay;`
  - ▶ - `(void)setNeedsDisplayInRect:(CGRect)unREct;`

9

# Pour dessiner...

Principes généraux

1. Obtenir un contexte dans lequel dessiner
2. Créer un chemin (à l'aide de traits, de courbes, ...)
3. Définir des couleurs de traits, des épaisseurs, ...
4. Tracer le trait ou remplir la forme

10

# Pour dessiner...

Dans le code

```
// Obtenir le contexte
CGContextRef context = UIGraphicsGetCurrentContext();

// Commencer le chemin
CGContextBeginPath(context);

// Se déplacer, tracer des traits
CGContextMoveToPoint(context, 200, 110);
CGContextAddLineToPoint(context, 260, 170);
CGContextAddLineToPoint(context, 260, 50);

// Refermer le chemin (seulement si on veut fermer la forme)
CGContextClosePath(context);

// Définir les couleurs de trait et de remplissage
[[UIColor greenColor] setFill];
[[UIColor redColor] setStroke];
```

**Jusque-là, rien n'a encore été dessiné à l'écran!**

11

```
// Obtenir le contexte
CGContextRef context = UIGraphicsGetCurrentContext();

// Commencer le chemin
CGContextBeginPath(context);

// Se déplacer, tracer des traits
CGContextMoveToPoint(context, 200, 110);
CGContextAddLineToPoint(context, 260, 170);
CGContextAddLineToPoint(context, 260, 50);

// Refermer le chemin (seulement si on veut fermer la forme)
CGContextClosePath(context);

// Définir les couleurs de trait et de remplissage
[[UIColor greenColor] setFill];
[[UIColor redColor] setStroke];

// Finalement dessiner
CGContextDrawPath(context, kCGPathFillStroke);
```

12