OLÁ IPHONE!

Entendendo o básico do **xcode**
por Glauco Primo

MINHA PRIMEIRA APLICAÇÃO: COPY AND PASTE

- Começando um projeto view-based no Xcode
- Com um projeto view-based os controllers da nossa view já são criados automaticamente

A ESTRUTURA DE PASTAS

- **Classes** – é onde irão ficar todas as classes que iremos editar e criar, parecido com C++
- **Other Sources** – onde está a nossa main, não devemos mexer aqui.
- **Resources** – é onde está a interface gráfica da nossa aplicação, toda aplicação só tem uma janela, mas podemos ter tantas views quanto se queira
- **Frameworks** – é onde vamos importar qualquer biblioteca ou framework que eventuamente precisaremos em nossos projetos
- **Products** – contém o nosso produto final, que no Xcode tem extensão .app e é nossa aplicação em si

COMEÇANDO A CODIFICAR

- **IBOutlet** – tipo que quando instanciado iremos passar a ver a variável por dentro do interface builder
- **IBAction** – quando o evento é disparado por dentro da aplicação os métodos desse tipo irão ser executados.

SINTAXE EM OBJECTIVE C

- **Declaração de tipos (Campo de Texto):**

  *IBOutlet UITextField *text1;*

- **Declaração de métodos:**

  -(IBAction)copy:(id)sender;

- **Obs.:** Pode ser óbvio, mas, todas as declarações devem ficar no ".h", e as implementações no ".m"

INICIANDO O INTERFACE BUILDER

- Para acessar o IB basta dar um duplo-clique no controller da view da sua aplicação, dentro de *Resources*.
- No Menu Superior, na aba **Tools**, encontraremos tudo que precisamos, *Library, Attributes Inspector e Connections Inspector*.

MODELANDO A APLICAÇÃO

- **Library** – onde visualizamos todos os objetos que estão à nossa disposição.
- **.xib** – onde está a view da nossa aplicação, o File's Owner que é onde faremos as conexões do IB com os controllers e o First Responder será explicado depois.
- **View** – é onde colocaremos os objetos que serão usados em nossa aplicação, podemos instanciar 1 view ou mais. Tudo que está dentro da view será visualizado na tela do nosso Iphone Simulator.

MODELANDO A APLICAÇÃO

- **Attributes Inspector** – onde podemos definir atributos para nossos objetos.
- **Connections Inspector** – onde vamos fazer as conexões com os controllers.
- As demais propriedades do inspector serão explicadas adiante.

DEPOIS DO MODELO

- Depois que modelamos todos os objetos em nossa view, estamos prontos pra fazer as conexões.
- Devemos ver o Connections Inspector do nosso File's Owner, onde todos os objetos instanciados deverão aparecer na aba Outlets, inclusive a view. Os métodos estarão todos na aba Received Actions.
- Basta fazer as conexões de cada instancia com o objeto dentro da view no IB.

CONTINUANDO A CODIFICAR

- Agora podemos implementar o que o botão da nossa aplicação exemplo faz, no caso é um botão que copia o texto de um field e cola em outro field, então basta que façamos:

```
- (IBAction)copy:(id)sender
{
        [text2 setText:[text1 text]];
}
```

RODANDO A APLICAÇÃO

- Para rodar aplicação basta clicar em ***Build > Build and Go*** no menu superior do Xcode.
- O Iphone Simulator deverá abrir com nossa aplicação já funcionando.

MUDAR A VIEW

- Para mudarmos de view em nossa aplicação é necessária a criação de outro .xib no caso o caminho será:
  - **File > new file > user interface > View XIB**
- Precisamos criar um botão de "trocar a view" tanto em nossa view principal quanto em nossa nova view e é claro precisamos fazer as conexões dos nossões butões às actions deles.
- Precisamos fazer as conexões de nossa nova view nos referencing outlets.

CRIANDO OS CONTROLLERS

- Agora precisamos criar os controllers da nossa nova view.
- Basta ir em **file>write class files**
- De o nome que preferir a sua nova view e não esqueça de criar também o .h do seu controller
- Na sua view principal não esqueça de dar um import do .h de sua nova view pois teremos de criar uma instancia da view secundaria em nossa view principal.

IMPLEMENTANDO O SWITCH

- Para implementar o switch view segue o código abaixo:

```
-(IBAction) switchViews {
        secondViewController *screen =
        [[SecondViewController alloc]
        initWithNibName:nil bundle:nil];
        screen.modalTransitionStyle =
        UIModalTransitionStyleCrossDissolve;
        [self presentModalViewController:screen
        animated:YES];
        [screen release];
}
```

IMPLEMENTANDO O SWICTH BACK

- Para implementar o switch back basta fazer:

```
- (IBAction) swithBack {
    [self dismissModalViewControllerAnimated:YES];
}
```