



MVVM

Objetivo da Aula

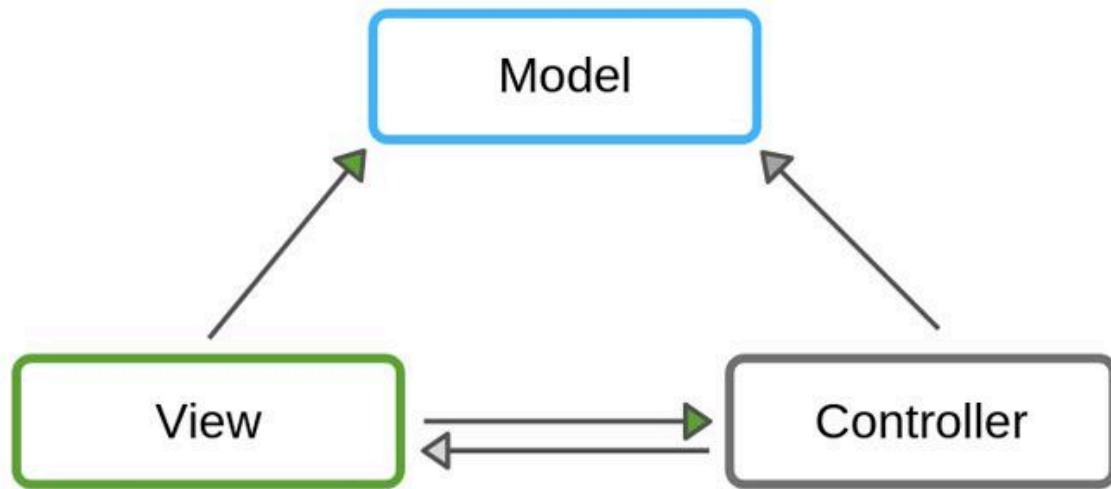
- Entendendo o que é MVVM
- Implementando MVVM




- Entendendo o que é MVVM

The background features a large, faint, light-gray diamond-shaped pattern composed of smaller squares. In the top-left and bottom-left corners, there are clusters of small triangles in red, white, and dark blue. In the top-right corner, there is a larger, more complex geometric pattern made of red, white, and dark blue triangles.

Relembrando...



Problemas MVC



Os problemas com aplicativos baseados em MVC começam com o passar do tempo. A base de código cresce e a complexidade também. Mesmo se você for um desenvolvedor experiente e estiver sendo cuidadoso, em algum momento você se verá consertando um inferno de callback em uma das centenas de View Controllers massivas.

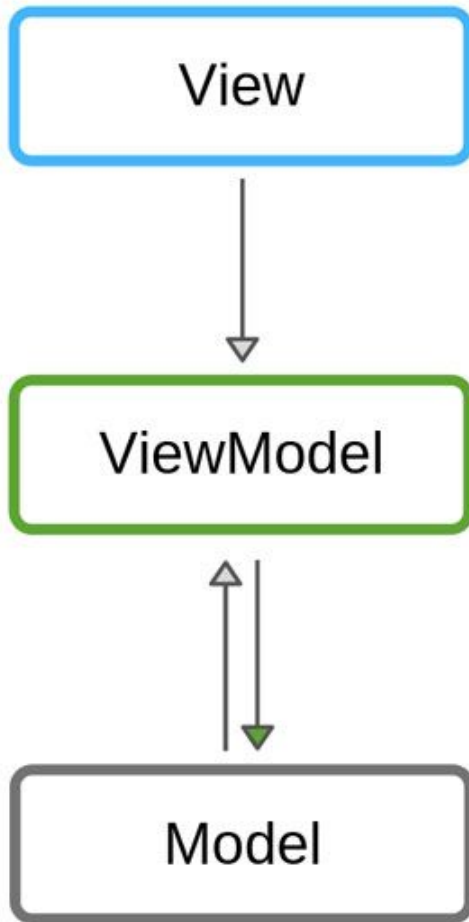
Você não terá ideia de onde todos os dados do aplicativo estão localizados e como você deve passá-los de uma parte do aplicativo para outra. Você não terá ideia de qual pedaço de código é responsável por quê e como chamar a lógica sem criar instâncias ocultas de algum View Controller. É muito fácil perder o controle de um aplicativo escrito com MVC conforme ele cresce.

The background features a light gray grid with a subtle dot pattern. In the top-left and bottom-left corners, there are decorative clusters of overlapping triangles in red, white, and dark blue. A similar pattern is located in the top-right corner.

MVVM

Definição

O MVVM é um pattern que foi criado em 2005, por John Gossman, um dos arquitetos do WPF(windows presentation foundation) e Silverlight na Microsoft. O MVVM, visa estabelecer uma clara separação de responsabilidades em uma aplicação



Definição

View – A responsabilidade da View é definir a aparência ou estrutura que o usuário vê na tela.

Definição

- * A View é um elemento visual

- * A View referencia a ViewModel através da instancia da viewModel

Definição

ViewModel – A responsabilidade da ViewModel no contexto do MVVM, é disponibilizar para a View uma lógica de apresentação. A View Model não tem nenhum conhecimento específico sobre a view, ou como ela implementada, nem o seu tipo. A ViewModel implementa propriedades e comandos, para que a View possa preencher seus controles e notifica a mesma, caso haja alteração de estado; seja através de eventos ou notificação de alteração. A ViewModel é peça fundamental no MVVM, por que é ela quem vai coordenar as interações da View com o Model, haja vista, ambos não terem conhecimento um do outro. E além de tudo isto, a ViewModel, também pode implementar a lógica de validação, para garantir a consistência dos dados.

Definição



- A ViewModel é uma classe não visual, que expõe para a View uma lógica de apresentação.
- A ViewModel é testável, independentemente da View ou Model.
- A ViewModel coordena as intenções entre a View e o Model.
- A ViewModel não referencia a View, na verdade não tem nenhum conhecimento sobre a mesma.
- A ViewModel expõe propriedade e comando, para que a View possa utilizar para preencher seus controles
- A ViewModel pode conter a lógica de validação







Definição

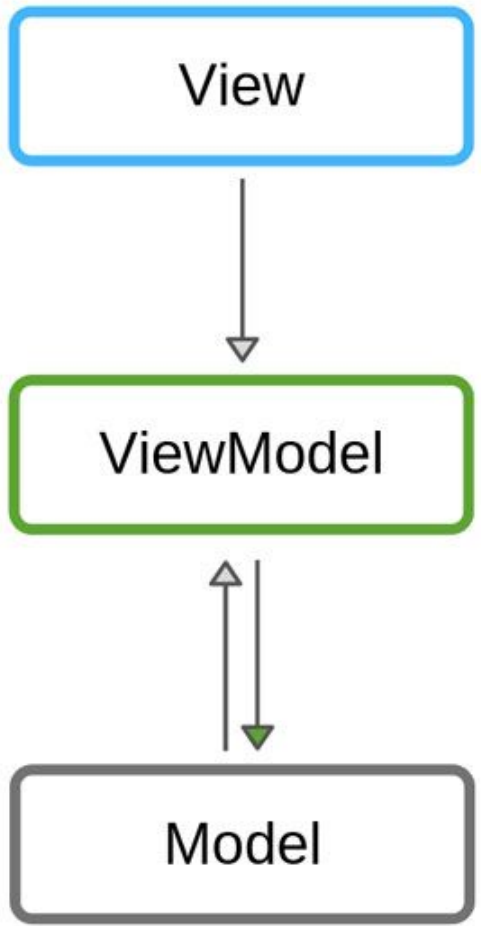
Model no MVVM, encapsula a lógica de negócios e os dados. O Modelo nada mais é do que o Modelo de domínio de uma aplicação

Problemas MVVM

No MVVM, você ainda terá muitos ViewModels que chamam métodos diferentes em diferentes partes do modelo em uma ordem aleatória. Por causa disso, você ainda nem sempre sabe qual lógica pertence ao ViewModel e o que é melhor mover para algum serviço. Você também não pode ter certeza de que um ViewModel não será excluído antes de receber um retorno de chamada no final de uma determinada tarefa. Ainda assim, com o MVVM, temos algum tipo de ordem em nosso aplicativo.



| | | | |
|---|---|--|---|
| ▼ | Model | | |
| |  Memes.swift | | A |
| ▼ | APIManager | | |
| |  APIManager.swift | | A |
| ▼ | ViewModel | | |
| |  MemesAPI.swift | | A |
| |  MemesViewModel.swift | | A |
| ▼ | ViewController | | |
| | ▼ | | |
| |  MemesTableViewCell.swift | | A |
| |  MemesViewController.swift | | R |



Quando usar uma arquitetura iOS?

MVC:

MVC é o melhor padrão de arquitetura para começar a aprender o desenvolvimento do iOS. Ele estrutura o fluxo de dados e a interação em seu aplicativo. Se você estiver construindo um aplicativo simples, este padrão de arquitetura iOS será ideal para você.

MVVM:

Use MVVM quando precisar transformar modelos em outra representação para uma visualização. Você pode usar um modelo de exibição para transformar uma data em uma string formatada por data, um decimal em uma string formatada por moeda.

MVVM é uma das melhores maneiras de reduzir os controladores de exibição massivos que requerem várias transformações de modelo para exibição.

Atividade

Atividade

- Criar uma nova branch no projeto do PI chamada *Arquitetura*
- Refatorar um *fluxo* para o padrão MVVM



Bora Coda

Links

Documentação MVVM- Documentação