Minicurso Knockout JS

Colaboradores
Érick Lopes, Mariana Pereira e Sandro Gaubert.

Índice da Apresentação

1. Apresentação da Biblioteca
2. Padrão MVVM
3. Aplicação [exemplos + prática]
4. Material de Apoio e Consulta
5. Encerramento

Introdução

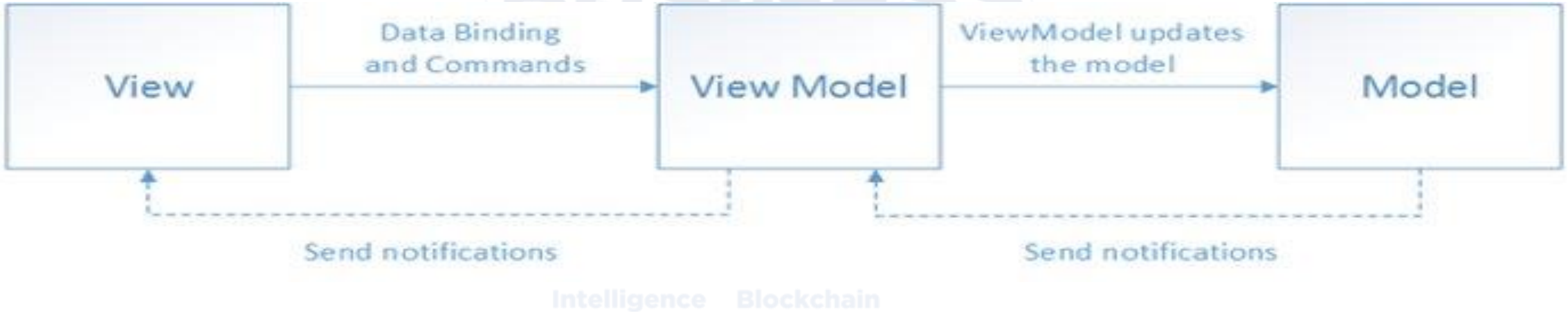
- Biblioteca de *Javascript* para desenvolvimento de Interfaces de Usuário (UI)
- Foi desenvolvida em Vanila.js e é pequena e leve (50 kb)
- É suportada pela maioria dos navegadores, mesmo os mais antigos
  - IE 6+, Firefox 3.5+, Chrome, Opera, Safari (desktop/mobile)
- Licença Massachusetts Institute of Technology (MIT)

Introdução

- Time Knockout (2010), onde estão Steve Sanderson, Ryan Niemeyer e Michael Best
- TKO (2017) - servirá de base para futuras versões da biblioteca, pretende tornar mais sólido o projeto knockout.
- Ótima Documentação

Padrão *Model-View-ViewModel* (MVVM)

- O padrão Model-View-ViewModel (MVVM) permite separar a lógica de negócios e apresentação de um aplicativo de sua interface do usuário (UI) de forma precisa. Permite resolver problemas de desenvolvimento e trocar facilmente a View para outra plataforma.



View, Model e ViewModel

- **Model**: Os dados armazenados da sua aplicação. Esses dados representam objetos e operações no domínio de sua empresa e são independentes de qualquer interface do usuário.
- **ViewModel**: Uma representação de código puro de dados e operações em uma interface do usuário.
- **View**: Uma interface do usuário visível e interativa que representa o estado do modelo de exibição. Ele exibe informações do modelo de exibição, envia comandos para o modelo de exibição (por exemplo, quando o usuário clica nos botões) e é atualizado sempre que o estado da exibição da view é alterado.

Aplicação

- Preparando nosso ambiente de desenvolvimento
  - Faça o clone do repositório https://github.com/erickLFLopes/minicursoKnockout
  - Importe o projeto em seu editor de texto preferido
  - Entendendo a estrutura de pastas utilizadas
- Observação: A base do projeto entregue ainda não possui a integração com a biblioteca, permitindo que o aprendizado ocorra com o acompanhamento da integração da lib.

Aplicação

- Passos iniciais
  - Entendendo a construção de uma view-model
  - Aplicação de bindings
  - Propriedades observáveis
  - Propriedades computadas
  - Funções (métodos da view model)

O Começo de Tudo

- Observable: Usa o padrão observer, implementa uma estrutura em que objetos observam modificações em outros objetos chamados de observáveis.
- Computed: Propriedade computável.
- PureComputed: Propriedade calcula sem dependência de variáveis globais.
- Data-bindings: Atributos data-bind é a forma que o Knockout faz declarativamente a associação de propriedades ViewModel com elementos do DOM.

Aplicação - Data-bind

- . As possibilidades trazidas pela biblioteca são:
  - Controle de aparência
  - Controle de fluxo
  - Manipulação de campos

Controle de aparência

- Entre os bindings de controle de aparência, estão:
  - **Visible -> Modifica diretamente o style display**
  - **Text -> Serve para modificar o texto de um campo, ou elemento**
  - **Html -> Recebe um elemento HTML para ser aplicado como filho do receptor**
  - **Css -> Utilizado para aplicar classes ao elemento**
  - **Style -> Utilizado para modificar propriedades de estilo**
  - **Attr -> Modificação de Atributos do elemento**

Atividades

- Com ajuda dos instrutores, use o binding adequado para modificar a cor do título da aplicação para uma cor de sua preferência.
- Adicione o texto com a data em itálico ao lado sub título cidade.
- Para cidades com o nome maior que 10 caracteres, mude o atributo class para texto-g
- [opcional] A cidade com o nome 'passo fundo' não deve ser exibida, crie um método para remove-lá.

Controle de fluxo

- Entre os bindings de controle de fluxo, estão:
  - **foreach -> duplica uma seção de marcação para cada entrada em um array**
  - **if -> Controla a presença/ausência de um elemento no DOM**
  - **ifnot -> Segue a lógica do if, porém nega a expressão utilizada**
  - **with -> Utilizado para criar contexto de objeto**

Atividades

- Percorra a lista de cidades e exiba elas de forma dinâmica.
- Liste os feriados da primeira cidade de nossa lista, neste ponto não nos preocuparemos em modificar os feriados para cada cidade da lista.
- Utilize um controlador para coibir a **inserção** da cidade onde o nome for Passo Fundo.
- [opcional] Modifique as cores do bloco de desgner do lado do nome das cidades

Manipulação de campos

- Entre os bindings de controle de aparência, estão:
  - **Click -> Binding que determina uma função para o evento de clique do elemento DOM**
  - **event -> Binding para que possamos determinar callbacks de eventos**
  - **submit -> Binding utilizado para atribuir ação na submissão de um formulário**
  - **enable -> Associado a input, selects e text-areas habilitado conforme o valor atribuído a ele (true ou false)**
  - **disable -> Lógica inversa do enable**
  - **value -> Vincula o valor de um elemento do DOM com uma propriedade em ViewModel**

Atividades

- Abra o modal da aplicação ao clicar no nome de uma cidade
- Capture o valor de data quando o campo data for modificado

Manipulação de campos

- Entre os bindings de controle de aparência, estão:
  - **textInput-> Funciona como o value, entretanto é acionado em todo evento**
  - **hasFocus -> Vincula o estado de foco do DOM a uma propriedade da viewModel**
  - **checked -> Associa controles checáveis com propriedades da viewModel**
  - **options -> Controla as opções que devem aparecer em uma lista suspensa**
  - **uniqueName -> Garante que o elemento tenha um nome não vazio**

- Apresente os feriados da cidade selecionada usando checked
- Pegue o valor de data em tempo real
- Caso o campo data esteja com foco, apresente um texto descrevendo que o limite de consulta é para o ano 2018

Renderizando Templates

- Os templates são uma maneira simples de construir estruturas de interface do usuário sofisticadas - possivelmente com blocos repetidos ou aninhados - como uma função dos dados do viewModel.

Desafio

- Faça com que o modal exposto no código apareça sempre que clicarmos em um dos feriados.
- Este modal deve apresentar a descrição do feriado no qual clicamos.

Material de Apoio e Consulta

- Site do projeto: https://knockoutjs.com/
- Repositório KO: https://github.com/knockout/knockout
- Repositório TKO: https://github.com/knockout/tko/
- MVVM: https://docs.microsoft.com/pt-br/xamarin/xamarin-forms/enterprise-application-patterns/mvvm
- XALM: https://docs.microsoft.com/pt-br/dotnet/framework/wpf/advanced/xaml-overview-wpf

Dúvidas Amiguinhos???


Design Thinking
Big Data
IoT
BUSINESS
TRANSFORMATION
INNOVATION
Digital
Artificial Intelligence
Blockchain