



André Paulovich

Growth Hacker - Arquiteto de Sistemas
Estrategista Digital, Agilista e eterno estudante
ION Sistemas

3 vezes Microsoft MVP - 2011 à 2014
MCP | MCTS | MCT | MCAD | MCSD.Net
andre.paulovich@ionsistemas.com.br



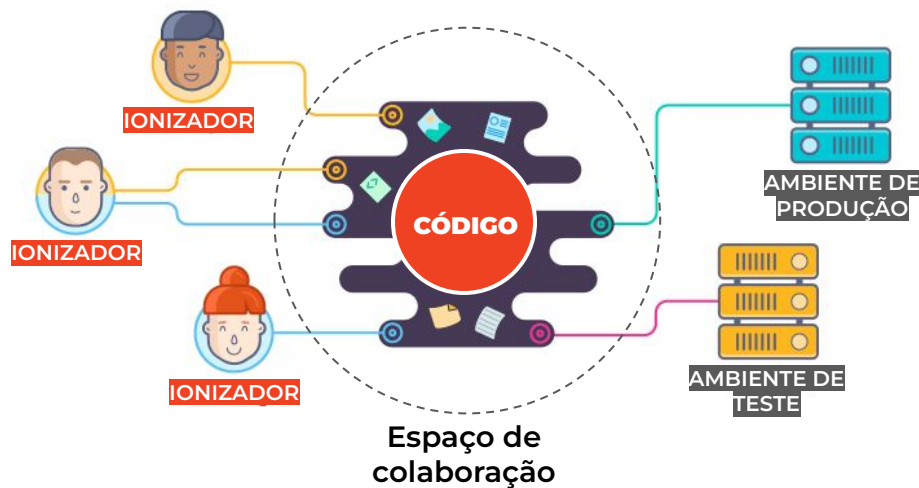
@andrepaulovich

DESENVOLVIMENTO WEB **COM ASP NET CORE**

Básico de GIT

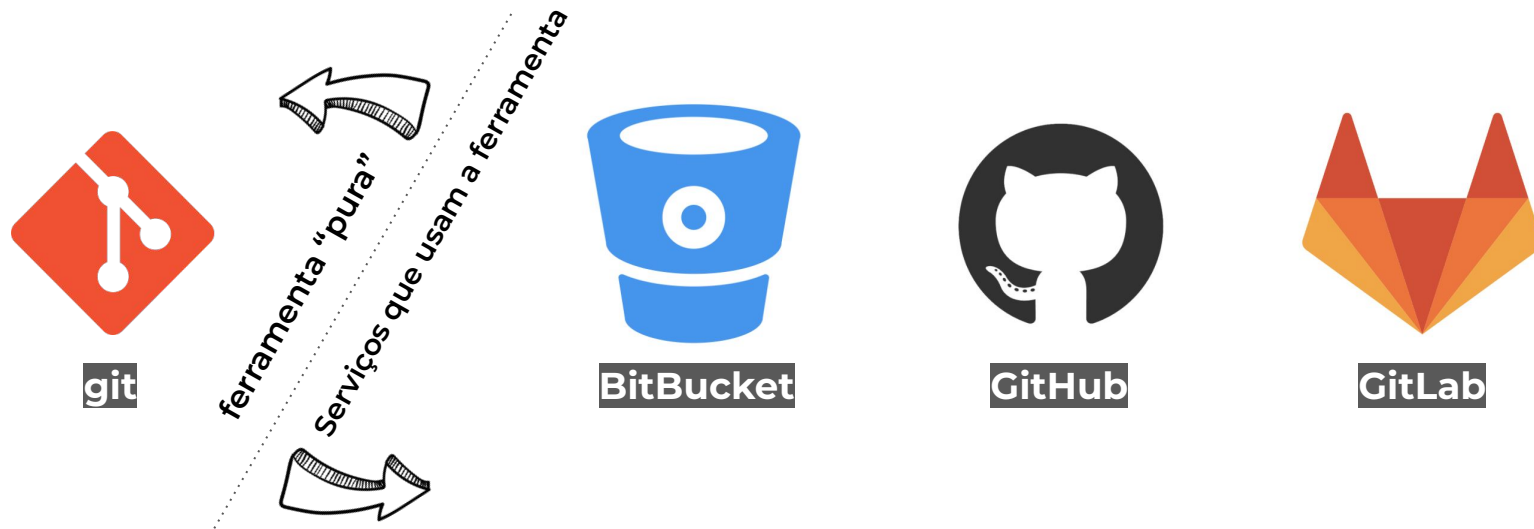
Colaboração

O trabalho de criar um software não é artesanal e feito de ponta a ponta por uma única pessoa, mas co-criado através de contribuições de diferentes pessoas. Por isso, é muito importante entender como se dá esta dinâmica de colaboração intensa.



GIT

Git é um sistema de **controle de versão open-source** e gratuito. Por meio dele, é possível observar a criação de histórico com versões das alterações realizadas no código-fonte dos projetos de desenvolvimento. Inclusive, foi criado por Linus Torvalds, que também é conhecido como o criador do **Linux**.



Fluxo de trabalho

Seus repositórios locais consistem em três "árvores" mantidas pelo git. a primeira delas é sua **Working Directory** que contém os arquivos vigentes. a segunda **Index** que funciona como uma área temporária e finalmente a **HEAD** que aponta para o último commit (confirmação) que você fez.



evolução do seu código

Baixar um repositório

Crie uma cópia de trabalho em um repositório local executando o comando

```
> git clone "https://caminho para o repositório"
```

Adicionar e confirmar mudanças

Você pode propor mudanças (adicioná-las ao Index) usando

```
> git add <arquivo>
```

```
> git add *
```

Este é o primeiro passo no fluxo de trabalho básico do git. Para realmente confirmar estas mudanças (isto é, fazer um commit), use

```
> git commit -am "comentários das alterações"
```

Agora o arquivo é enviado para o HEAD, mas ainda não para o repositório remoto.

Enviando as alterações

Suas alterações agora estão no HEAD da sua cópia de trabalho local. Para enviar estas alterações ao seu repositório remoto, execute

```
> git push origin master
```

Altere *master* para qualquer ramo (branch) desejado, enviando suas alterações para ele.

Se você não clonou um repositório existente e quer conectar seu repositório a um servidor remoto, você deve adicioná-lo com

```
> git remote add origin <servidor>
```

Agora você é capaz de enviar suas alterações para o servidor remoto selecionado.

Atualizando e mesclando

Para atualizar seu repositório local com a mais nova versão, execute

```
> git pull
```

Na sua pasta de trabalho para obter e fazer merge (mesclar) alterações remotas.
para fazer merge de um outro branch ao seu branch ativo (ex. master), use

```
> git merge <branch>
```

Em ambos os casos o git tenta fazer o merge das alterações automaticamente. Infelizmente, isto nem sempre é possível e resulta em conflitos.

Atualizando e mesclando

Você é responsável por fazer o merge destes conflitos manualmente editando os arquivos exibidos pelo git. Depois de alterar, você precisa marcá-los como merged com

```
> git add .
```

Setup do Ambiente

Setup do ambiente windows

Antes de começarmos a abordar os temas de programação propriamente dita, precisamos fazer um setup básico do nosso “ambiente” de desenvolvimento. Eu gosto muito de executar esse processo usando o máximo possível de ferramentas de linhas de comando, para garantir que estamos de fato realizando o processo, exatamente igual. Evitando assim dúvidas sobre versões e configurações dos programas.

Para isso vamos instalar o **Chocolatey**.



1. Acesse o site: <https://chocolatey.org/install>
2. Abrir o PowerShell como 'administrador' e seguir os passos que estão descritos na página
3. PRONTO!!!

Abra um “prompt de comando” como administrador e digite “**choco**”, a versão deve aparecer. Se sim, **TUDO OK!**

```
> choco
```

```
λ choco  
Chocolatey v1.2.0  
Please run 'choco -?' or 'choco <command> -?' for help menu.
```

Instalando o VS CODE e o AspNet Core

Utilizando o **Chocolatey**, vamos instalar o **Visual Studio Code** e o **AspNet Core**.

Para isso, abra um “prompt de comando” como administrador e digite:

- Instalar o Visual Studio Code:

```
> choco install vscode
```

- Instalar o AspNet Core:


```
> choco install dotnet-sdk --version=7.0.100
```

Neste momento, estou utilizando a versão **“7.0.100”**, que é a versão mais atual.


```
> dotnet --version
```

Extensões do Visual Studio Code


Abra o **Visual Studio Code** e instale as seguintes extensões (elas serão necessárias para turbinar nosso trabalho):



C# v1.25.2
Microsoft | 19,269,418 | ★★★★★ (418)
C# for Visual Studio Code (powered by OmniSharp).
Set Color Theme Disable Uninstall ⚙️
This extension is enabled globally.



VS Sharper for C# v0.2.0
eService Online | 43,260 | ★★★★★ (4)
C# IDE, Code Generation and Refactoring for VS Code
Disable Uninstall ⚙️
This extension is enabled globally.



IntelliCode v1.2.29
Microsoft | 25,075,069 | ★★★★★ (84)
AI-assisted development
Disable Uninstall ⚙️
This extension is enabled globally.

Setup do ambiente Linux

Execute estes comandos no **#!/bin/bash**

```
mkdir -p $HOME/installbootcamp
```

```
cd $HOME/installbootcamp
```

```
sudo apt update
```

```
sudo apt install -y unzip wget curl git zsh terminator build-essential docker.io
```

```
wget -c https://dotnet.microsoft.com/download/dotnet/scripts/v1/dotnet-install.sh
```

```
sudo chmod +x ./dotnet-install.sh
```

```
./dotnet-install.sh --channel 7.0
```

```
sudo ln -s $HOME/dotnet/dotnet /usr/local/bin/dotnet
```

```
dotnet --version
```

```
curl -O https://az764295.vo.msecnd.net/stable/97dec172d3256f8ca4bfb2143f3f76b503ca0534/code_1.74.3-1673284829_amd64.deb
```

```
sudo dpkg -i code_1.74.3-1673284829_amd64.deb
```

```
sudo apt install -f -y
```

```
code --install-extension ms-dotnettools.csharp --force
```

```
code --install-extension visualstudioexpteam.vscodintellicode --force
```

```
code --install-extension eservice-online.vs-sharper --force
```

```
sudo apt install sqlite3
```

Setup do Projeto

Criando o projeto

Uma vez instalado o AspNet Core, temos acesso ao “dotnet CLI” que reúne uma coleção de comandos para nos ajudar a trabalhar com o SDK. <<https://learn.microsoft.com/pt-br/dotnet/core/tools/>>

Neste curso, vamos criar um WEB APP de **gerenciamento de tarefas**. Portanto por padrão, vamos criar uma pasta chamada ‘Tarefas’ e dentro dela uma pasta específica para os códigos, chamada de ‘src’:

```
> mkdir Tarefas  
> cd Tarefas  
> mkdir src  
> dotnet new sln
```

Note que irá criar a solução com o mesmo nome da pasta onde o comando está sendo executado: **‘Tarefas’**.

Vamos então, abrir o Visual Studio Code no diretório "Tarefas" e perceber que na primeira vez que abrirmos a solução o Visual Studio Code baixará as dependências para trabalhar com **C#**.

Configurações do SDK

O Arquivo “**global.json**” deve ficar na raiz do projeto, no mesmo nível da solução. Para criar o “global.json” com o último “sdk” instalado. Para isso, abra um “prompt de comando” como administrador e digite:

```
> dotnet new globaljson --sdk-version 7.0.100
```

Neste momento, estou utilizando a versão “**7.0.100**”, que é a versão mais atual do SDK do .Net.

```
{  
  "projects" : ["src"],  
  "sdk": {  
    "version": "7.0.100"  
  }  
}
```

Edite o arquivo da seguinte maneira e inclua linha "projects" com a declaração da pasta 'src'.

Criando o projeto MVC

O “dotnet CLI” possui um gerador de projeto MVC, que nos entrega uma estrutura básica de arquivos com uma navegação inicial que é excelente para conhecermos mais sobre a arquitetura deste tipo de solução.

```
> cd src  
> mkdir Tarefas.Web  
> cd Tarefas.Web  
> dotnet new mvc
```

Apesar de ter criado o projeto "**Tarefas.Web**" na pasta "src", esse projeto ainda não é conhecido pela solução. (o projeto foi criado usando o mesmo nome da pasta onde o comando foi executado)

Para adicionar o projeto que criamos agora à solução, então execute o comando:

```
> dotnet sln add ./src/Tarefas.Web/Tarefas.Web.csproj
```

Executando o projeto pela primeira vez

Para executar o projeto e vê-lo funcionando, precisamos estar na pasta “Tarefas.Web”, e então executar:

```
> dotnet run
```

Aguarde alguns segundos até que o build seja executado e a aplicação iniciada.

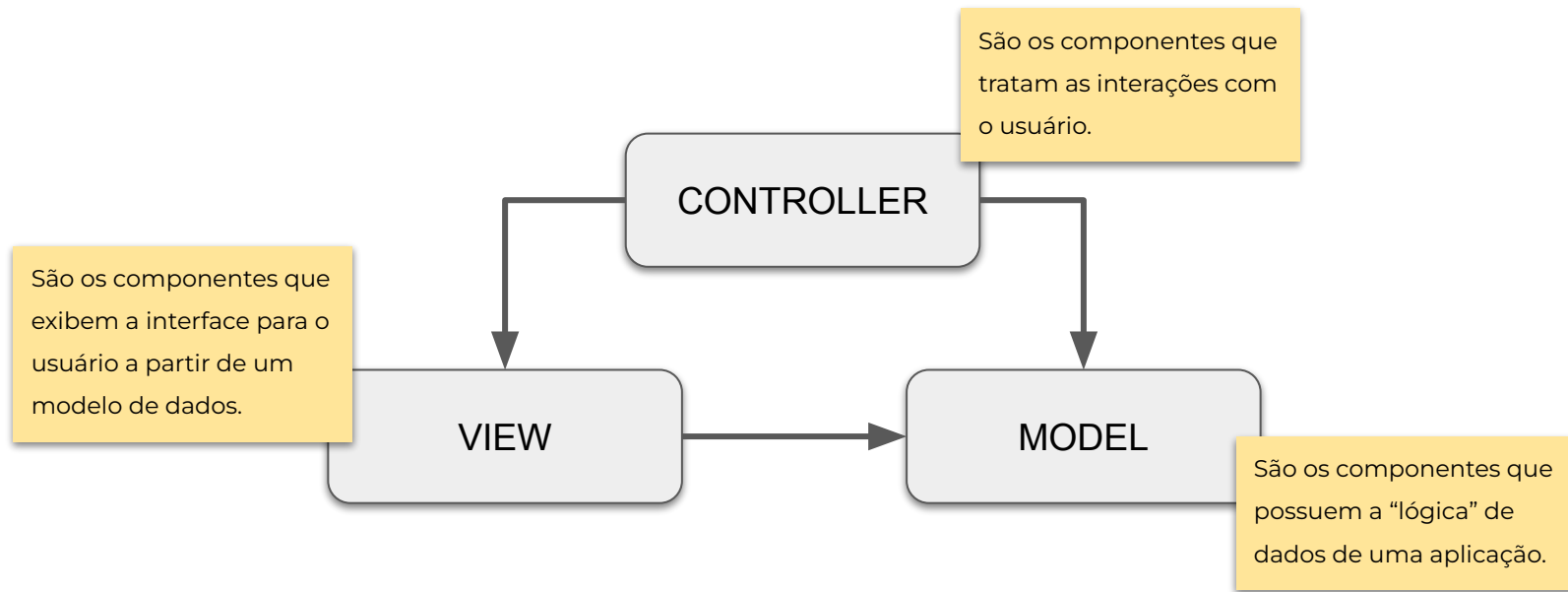
```
Compilando...
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5240
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
```

Então, basta abrir o navegador e acessar a url <https://localhost:{porta}> // porta é um número randômico gerado durante o build

Conceitos importantes

Convenções do AspNet MVC

Antes de seguirmos para o desenvolvimento da nossa APP, vamos comentar um pouco sobre algumas convenções da arquitetura que foi implementada no AspNet MVC.



Convenções do navegação

As URLs de navegação de uma aplicação AspNet MVC estão relacionadas aos controladores que serão acionados pelo sistema através de uma convenção. Isso quer dizer, que ainda que não exista uma ligação explícita entre os componentes URL, View e Controller... a “engine” irá sempre utilizar esta convenção para determinar os componentes que irão processar a requisição da seguinte forma:

baseurl/{controller=home}/{action=index}/{id?}

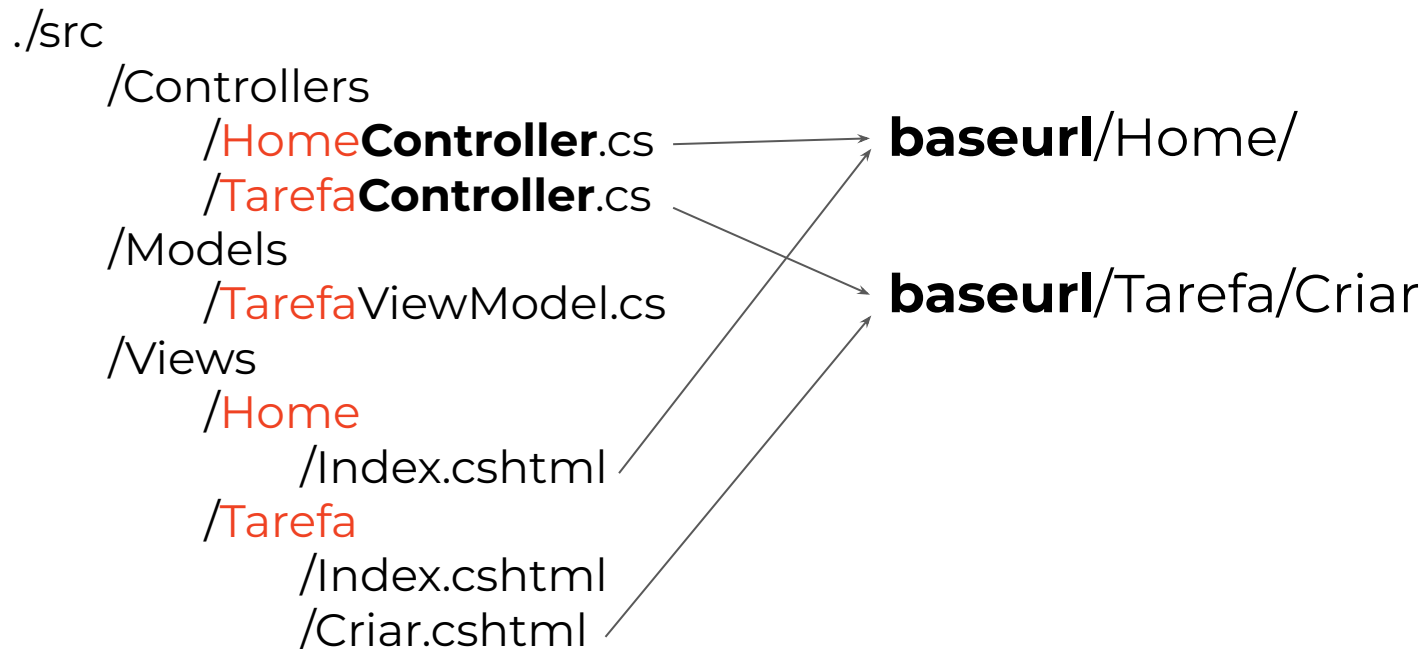
baseurl/tarefa/criar

baseurl/tarefa/editar/1

baseurl/tarefa/excluir/1

Convenções de arquivos

Para entender a estrutura de arquivos de uma aplicação MVC, é importante saber que existe uma série de convenções que envolvem a nomenclatura dos arquivos e pastas.



Rascunho de nossa aplicação

○○○

Login

Minhas Tarefas

Informe seu e-mail e senha para entrar.

e-mail

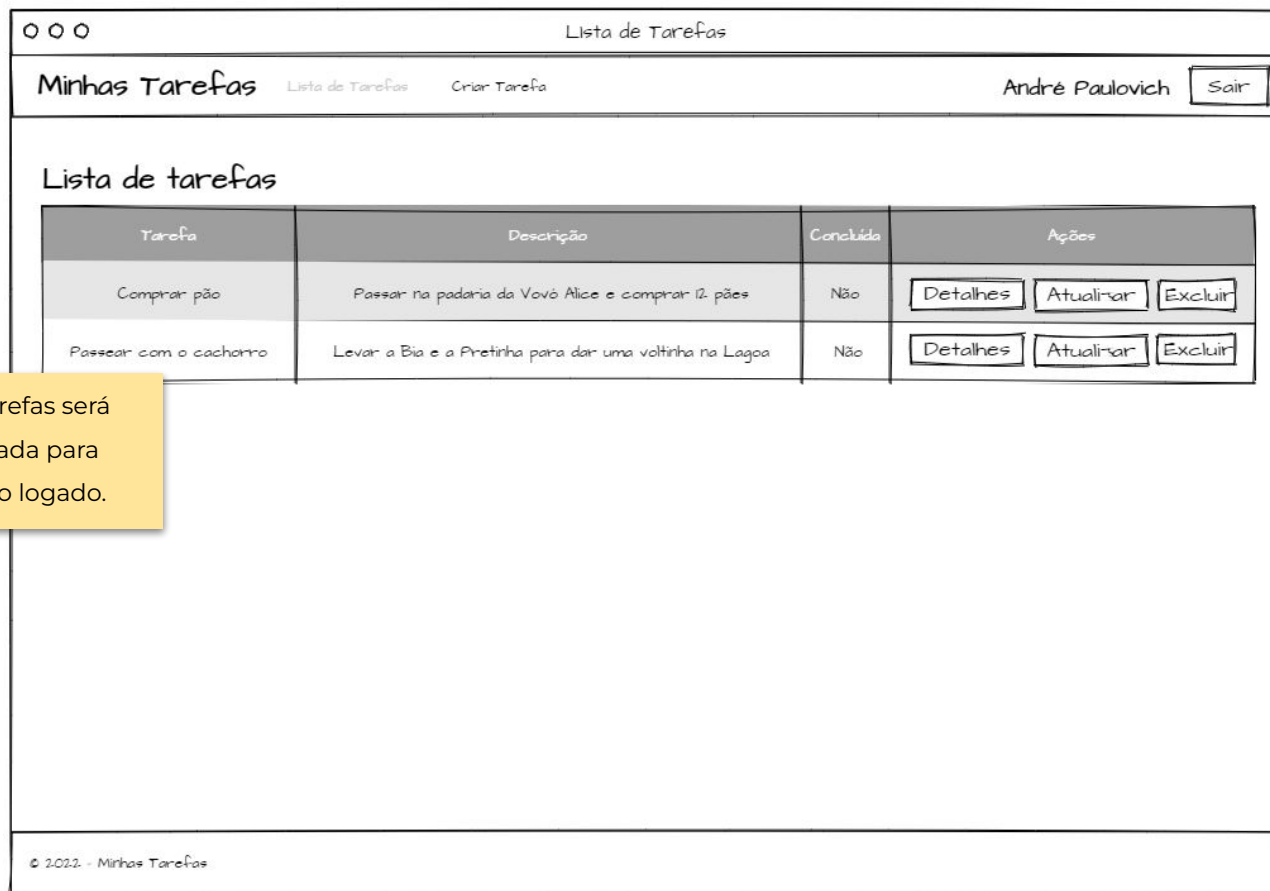
senha

Entrar

Para entrar no sistema, o usuário precisará informar o email e a senha.



Uma vez logado, o sistema vai mostrar o nome do usuário no topo da página.



A lista de tarefas será individualizada para cada usuário logado.

0 0 0

Criar Tarefa

Minhas Tarefas

Lista de Tarefas

Criar Tarefa

André Paulovich

Sair

Criar Tarefa

Título

Passear com o cachorro

Descrição

Levar a Bia e a Pretinha para dar uma voltinha na Lagoa

☐ Sim
 ☒ Não

Salvar

Voltar

© 2022 - Minhas Tarefas

Sempre que uma tarefa for cadastrada, deve ficar relacionada ao usuário logado no momento.

Bora pro código!