

## Integração com o waze

### Transcrição

[00:00] A próxima funcionalidade que nós vamos implementar no nosso app é chamar aplicativos externos a partir do nosso app.

[00:08] Uma feature legal pra nós implementarmos no nosso app, por exemplo, seria conseguirmos localizar os nossos alunos no Waze e traçar uma rota a partir de onde estamos até onde está cadastrado o endereço deles. Nós vamos aprender a fazer isso agora.

[00:23] Vamos dar uma olhada, se eu fizer um long press, ele vai abrir aqui um action sheet que nós criamos. Nós precisamos criar um novo botão pra conseguirmos chamar essa funcionalidade do Waze. Vamos lá então?

[00:33] Aqui no arquivo “MenuOpcoesAlunos” eu vou criar no nosso enum mais um case, que vai ser “waze”. Agora precisamos criar um novo botão, então “let waze = UIAlertAction”, eu vou inicializá-la passando o título, nós podemos chamar essa funcionalidade, por exemplo, de “localizar no waze”. O estilo dela vai ser o “default” e aqui na closure de completion nós passamos a nova propriedade no enum que nós criamos, que é “waze”.

[01:08] Depois que nós criamos o botão, nós precisamos adicioná-lo no menu, então “menu.addAction”, aqui nós passamos “waze”. Nós já temos um novo botão no nosso action sheet. Agora nós podemos voltar no View Controller pra implementar esse novo case. Embaixo aqui do break eu vou criar um “case.waze” e aqui vai ser onde vamos implementar toda a nossa lógica pra abrir essa funcionalidade.

[01:35] Primeira coisa que nós temos que pensar, será que todo usuário iOS tem o Waze instalado? Tanto no seu iPhone ou no seu iPad? Certamente não. Embora o Waze seja um app bem legal e bem útil, mas não é obrigado todo mundo a tê-lo instalado no iOS, então pode ser que algum usuário do nosso aplicativo não tenha ele instalado, aí pode ser que ocorra um crash se nós não fizermos uma verificação. O primeiro passo é verificar se o usuário tem o Waze e se nós podemos acessar essa funcionalidade. Então vamos lá.

[02:06] Assim como nós fizemos na chamada telefônica, nós utilizamos a classe UIApplication, nós vamos usá-la novamente pra fazer essa verificação. Vou colocar aqui um “if UIApplication.shared.canOpenURL”. Ele espera uma URL, então nós vamos criar uma URL e vamos escolher o método construtor dela, onde nós passamos uma string, que é esse aqui.

[02:32] Pra verificarmos se o Waze está instalado, o que eu vou fazer? “waze://”. Se ele estiver instalado e o iOS puder abrir, ele vai entrar no nosso “if”, e agora sim nós precisamos pensar como vamos fazer essa implementação.

[02:48] O primeiro passo é analisarmos o nosso formulário, o que nós temos que seria útil pro Waze localizar o nosso aluno? Vamos abrir aqui. Certamente é o campo endereço, onde nós cadastramos aqui “MASP, São Paulo”.

[03:02] Só que tem aqui um ponto importante que nós temos que entender, o Waze ou qualquer outro aplicativo de mapa, não é apenas com uma string, com um endereço que ele consegue fazer essa localização, na verdade eles utilizam coordenadas geográficas, que nada mais são do que latitude e longitude. Através dessas duas informações ele consegue localizar o aluno no mapa pra nós.

[03:26] Só que nós não temos a latitude e longitude, nós temos o endereço. Então precisamos achar uma forma de converter o endereço, que é a informação que nós temos, em latitude e longitude, aí fica fácil pro Waze conseguir abrir pra nós a localização dos nossos alunos. Então vamos lá.

[03:42] Para fazermos essa conversão, eu vou criar uma nova classe, que vai se chamar localização, então eu vou vir em novo arquivo e vou chamar de “Localizacao”. Vou dar um next e um create. Bacana, agora eu já posso criar um método aqui, então “func converteEnderecoEmCoordenadas”.

[04:09] Para convertermos o endereço em coordenadas, nós precisamos fazer uso de uma classe que se chama CLGeocoder, então vamos criar uma constante “let conversor = CLGeocoder”. Só que repare que eu estou digitando “CLGeocoder” e não está aparecendo para mim a referência dessa classe. Isso acontece porque o CLGeocoder faz parte de um pacote, de um framework nativo do iOS, que se chama Core Location, então nós precisamos importar o Core Location para termos acesso a essa classe.

[04:45] Aqui em import eu vou dar um “import CoreLocation”, eu tenho acesso a classe CLGeocoder. Instanciei-a. Agora vou chamar o “conversor.”, ele tem vários métodos construtores aqui, o que nós precisamos é um que nós passemos o endereço e ele retorne as coordenadas, que é esse primeiro carinha, “geocodeAddressString”, então eu vou utilizá-lo. Vou dar um enter.

[05:12] Repare que aqui ele pede o endereço e ele tem uma closure de completion, que é o que ele devolve pra nós, a localização. Só que eu não tenho o endereço do aluno aqui, eu tenho que pedi-lo através do nosso método, então aqui na assinatura do método eu vou pedir o endereço, “endereco”, ele é do tipo “String” e eu vou utilizá-lo aqui, “endereco”. Aqui ele é uma closure, onde ele vai nos retornar duas informações. Vamos analisar?

[05:40] A primeira, o que nós podemos notar aqui? Que é uma lista, um array. Como eu sei disso? Através das chaves aqui, então ele é um array optional de “CLPlacemark”. Esse “CLPlacemark”, é o tipo de variável que guarda essa localização, ele é apenas isso. E aqui também ele nos traz o “Error”, que também é optional, que pode ser que tenha, pode ser que não tenha. Eu vou dar o nome para essa variável de “listaDeLocalizacoes” e aqui “error” mesmo.

[06:14] Como ele é optional, nós precisamos extrair o valor dessa lista, e como ele é uma lista, eu vou pegar apenas o primeiro objeto que ele encontrar, então vou fazer o seguinte, “if let localizacao = listaDeLocalizacoes.first”, como ele é uma lista eu vou pegar o primeiro. Agora sim eu extraí o valor dessa lista e eu tenho acesso a localização, que é do tipo “CLPlacemark”.

[06:41] Só que eu preciso ter acesso a essa localização no meu View Controller, é aqui que eu preciso ter acesso pra conseguirmos fazer as implementações que nós precisamos. Como eu posso devolver essa informação pra lá? Eu posso também criar uma closure aqui, onde eu passo essa localização que nós pegamos.

[06:59] Vou criar aqui, vai chamar “local”. O que nós precisamos passar? Precisamos passar essa localização que extraímos no “if let”, vamos fazer isso, vai ser “local”, do tipo “CLPlacemark”, que é o tipo que ele retorna para nós. E aqui então nós podemos passar.

[07:20] Vou fazer o seguinte, aqui nós chamamos de “local”, não foi? Então vou colocar aqui “local”, que é essa closure e vou abrir aqui, e ele espera que nós passemos alguma coisa do tipo “CLPlacemark”, o que nós temos desse tipo? A localização do “if let”, então vou passar aqui a localização.

[07:40] Agora que nós já criamos o método e está tudo certo, nós precisamos utilizá-lo aqui no View Controller. Dentro do “if” eu vou fazer o seguinte, eu vou chamar a classe “Localizacao” que nós criamos, instanciei e vou dar um “.converteEnderecoEmCoordenadas”. Só que o endereço que nós temos é optional, então nós precisamos extrair esse valor.

[08:04] Vou criar aqui um “guard let enderecoDoAluno” é igual ao “alunoSelecionado”, que nós já temos acesso “.endereco”. Repara que ele é optional mesmo. Se não der certo, o que eu vou fazer? Vou pôr aqui o “else” e vou dar um “return”. Se não tiver endereço, não faz sentido nós tentarmos converter alguma coisa em coordenadas.

[08:31] Agora que eu tenho o endereço do aluno, eu posso passá-lo aqui, “enderecoDoAluno” em um local, a closure que nós criamos, que é a “localizacaoEncontrada”. Agora que nós já conseguimos converter o nosso endereço em coordenadas geográficas, nós precisamos entender como nós acessamos a latitude e a longitude.

[08:53] É da seguinte forma, eu posso vir e pegar o “localizacaoEncontrada.location.coordinate.latitude”, nós acessamos a latitude dessa forma. E a longitude do mesmo jeito, o nome da nossa variável, “localizacaoEncontrada.location.coordinate.longitude”. É dessa forma que nós acessamos a latitude e a longitude.

[09:22] Só que pra abrir o Waze, nós precisamos criar uma URL em que nós passemos essa coordenada e essa coordenada latitude, é do tipo “CLLocationDegrees”, é esse tipo estranhinho aqui. Nós precisamos converter isso numa string, pra depois gerarmos a URL. Então vamos lá. Vou criar uma constante “let latitude”, vou criar uma string.

[09:50] Pra convertermos algum valor numa string nós podemos usar essa propriedade “describing” da string. E nós pegamos toda essa linha que nós acessamos a latitude e a colocamos aqui dentro. Assim nós temos uma constante que vai guardar pra nós a latitude.

[10:15] Agora podemos fazer o mesmo com a longitude, então “let longitude” é do tipo “String”. Nós passamos esse “describing” pra ele converter. Agora o que nós vamos fazer? Colar a longitude, então “.longitude”.

[10:37] Esse location é optional, só que se ele retornar pra nós, significa que ele encontrou alguma localização, então eu vou pôr aqui “!” só pra ele não retornar como optional. Aqui não tem perigo porque nós já estamos tratando, em localização, nós estamos fazendo um “if let”, estamos pegando o primeiro valor, e se existir, certamente ele vai ter a location, que é a propriedade que nós vamos utilizar.

[11:03] Agora que nós temos a latitude e a longitude, nós precisamos criar a URL que vamos utilizar para passar pra esse “UIApplication” abrir o Waze pra nós. Vamos lá, “let url”, primeiro vamos fazê-la do tipo “String”, que nós vamos concatenar todos os valores.

[11:18] Para abrirmos o Waze, o que nós precisamos fazer? “waze://”, aí nós precisamos passar primeiro a latitude, então eu vou passar aqui esse “l=” e aqui eu vou interpolar a string, “(latitude)”. Em seguida, eu vou colocar uma vírgula sem dar espaço nenhum e nós vamos passar a longitude, então “longitude”.

[11:51] Além disso, nós podemos pôr uma propriedade nessa URL pra pedir pro Waze já iniciar a navegação pra nós. Então vou pôr aqui um “&” e vou pôr a propriedade “navigate=yes”. Então nós criamos a URL.

[12:09] Finalmente nós podemos pedir pra classe UIApplication abrir o aplicativo pra nós. Então “UIApplication.shared.open(url:”, só que o que nós temos aqui é uma string e ele pede uma URL. O que nós temos que fazer? Converter a string pra URL. No método construtor da URL eu vou pôr “string:” e aqui eu passo a URL que nós criamos.

[12:43] Ele tem um dicionário, eu não vou utilizar nenhuma dessas opções, então eu vou criar um dicionário vazio. E na closure de completion eu não vou fazer nada, eu posso passar aqui “nil” mesmo.

[12:54] Criei a latitude, criei a longitude, criei URL e converti a URL, que era string, pra URL mesmo. Agora sim nós podemos testar. Como nós tínhamos comentado, algumas funcionalidades nós não conseguimos testar no simulador, temos que testar realmente em um iPhone. Como eu já estou com meu iPhone aqui plugado, eu vou rodar a nossa implementação para nós testarmos nele.

[13:19] Ainda vai faltar uma configuração, mas eu vou mostrar pra vocês qual a mensagem que ele mostra aqui no console pra podermos identificar esse problema. Vou rodar aqui no iPhone e nós já vamos testar.

[13:34] Ele já gerou o build aqui no iPhone, vou fazer um long press em cima do nosso primeiro aluno e vou clicar em “localizar no waze”. Repara que não aconteceu nada. Vamos dar uma olhada no console pra ver o que ele está pedindo?

[13:48] Ele fala que não consegue abrir a URL, ele falhou porque a URL “waze://”, ele precisa permitir que ele faça uma consulta no schema. Na verdade, o que nós precisamos fazer é adicionar uma propriedade nesse arquivo “Info.plist”. Eu estou até com o link da documentação, onde nós utilizamos o Waze. Aqui ele mostra vários exemplos como implementa no Android, no iOS. Aqui está em object vc, mas é bem parecido.

[14:17] E aqui embaixo ele fala o seguinte, quando você for compilar no iOS SDK 9 ou o posterior, você precisa incluir esses componentes no arquivo “Info.plist”. É isso que falta nós fazermos.

[14:31] Eu vou copiar exatamente do jeito que está aqui, vou voltar pro projeto. Esse link eu vou deixar na hora dos exercícios, então vocês podem acessá-lo lá. Copiei esse cara aqui, vou voltar no nosso projeto. Aqui em “Info.plist”, botão direito e vou abri-lo como “Source Code”, ele é parecido com XML. E embaixo desse array eu vou dar um enter e vou colar aquelas propriedades que nós pegamos lá na documentação, onde ele mostra como implementa o Waze no nosso app. Já cole.

[15:04] Pra eu ter acesso ao “Info.plist” daquela forma de novo, porque se eu clicar aqui e voltar lá, ele está como XML. Pra mostrar aquela visualização de novo, “Open As”, “Property List”, ele mostra do jeito que já estava.

[15:19] Agora que nós já colocamos essa configuração, vamos testar novamente. Vou rodar o app de novo aqui no meu iPhone. Vamos testar? Vou fazer o long press novamente em cima do João e vou clicar em “localizar no waze”. Cliquei. Ele já abriu aqui o Waze, através do nosso app e ele está procurando a rota e ele encontrou.

[15:46] Então a partir de onde eu estou, que é na Caelum aqui de São Paulo, ele traçou uma rota até o MASP, lá na Avenida Paulista pra nós irmos. Vou clicar em “Ir agora”. Ele mostra certinho para nós a rota, até chegar onde nós precisamos.

[16:03] Essa é uma das integrações que dá pra fazer entre um aplicativo que nós estamos desenvolvendo e é bem útil, nós utilizamos um app bem famoso, que tem toda essa estrutura, pra utilizarmos no nosso aplicativo. Então essa é uma das formas de se utilizar aplicativos externos a partir do nosso app.