TEST FOR IOS “UI DATA BINDING” DEVELOPER

Purpose of the test

We would like to test your skills as an iOS “UI data binding” developer

What is an iOS “UI data binding” developer

When we develop iOS apps in DigitalMind, we always divide the work into several specialties. Below are some of the specialties we have in DigitalMind when developing iOS apps

- iOS UI developer: A specialist who goes above and beyond to develop a pixel prefect UI that meets the client expectations (colors, fonts, font sizes, responsive design, spacing between elements, animations, third party UI components and custom controls etc.). We call this an iOS UI developer.
- iOS "UI data binding" developer: A specialist who devises an efficient and robust code that binds all the code from the UI developer up with the logic of the app (write algorithms, make logic that shows things from the database on the page, make API calls, third party frameworks integration, application architecture etc.). We call this specialty an iOS “UI data binding” developer.

In our company, these two above specialties are always made by two different persons. It is never the same person doing the UI development and the UI data binding. The reason for this is that our customers have a very high demand for the apps we build. So we want our colleagues/employees to be experts (and train) in one field instead of multi fields.

How to do this test

- You shall pretend you are an iOS “UI data binding” developer in our company. So you must do what a normal iOS “UI data binding” developer in our company does. We have made a small realistic test case that, includes all the UI. Your job is to write a robust code that binds all the UI components and works as expected (one page in an app and a calendar function). The name of the fictive company you shall make the solution for is called “CarFit”. Here are what you shall do regarding making the test:
    1. Read this document thoroughly
    2. Download the files you shall use for this solution (https://1drv.ms/u/s!Ap3qEzIs1WzjgkZ973oneHdlNl-5?e=1HwJe8)
    3. Make your solution for the test case
    4. Send files to us (on WeTransfer.com – it is 100% free to use)
        - A XCode project with your solution (it must be possible to run the solution out of the box)
    5. We will look thoroughly on your test results. We will give you feedback on your test. If you are one of the best in the test you will get a job at our company.

Links to file you MUST download

- We have made a XCode project, that you shall use to make your test in. The JSON you require is already added in the project. We have also included the JSON as a separate file, if you want to modify the same. You can download both the files via the link below.

https://1drv.ms/u/s!Ap3qEzIs1WzjgkZ973oneHdlNl-5?e=1HwJe8


CASE
DESCRIPTION

Introduction to case

This is a fictive case about the company CarFit.

Explanation of words

- CarFit = CarFit is a company that does carwash manually for their clients at their home address.
- CarFitClient = A person who buys car wash from CarFit. He can subscribe for a carwash each month or book a carwash when needed, by calling CarFit's office.
- CarFitPlanner = An administrative employee working at CarFit. His job is, to plan visits for a CarFitClient so they can get their carwash. He takes orders by phone and schedules a CarFitWashers Gmail calendar so the CarFitWasher can see a list of car washes for a day.
- CarFitWasher = The person doing the actual carwash at the client's home address. He drives around in his van with washing equipment and looks in his Gmail calendar for information about CarFitClient he needs to visit during the day.

Current situation for CarFit

Case description

- CarFit has a successful business going, doing carwash manually for their CarFitClients at their home address. Currently, they have more than 5,000 CarFitClients. Now they require a mobile app (made in iOS) to help them manage their business better. This app will be used by their CarFitWashers and will show a list of carwashes for a specific day. The CarFitWashers is then able to see assigned carwashes and get information about the CarFitClient like address, phone, driving assistance, etc.

Carfit's current software stack

- CarFit has a web application system right now where all their CarFitClients are listed. CarFit uses the CarFitClients email, address, name, mobile phone.
- All the CarFitWashers are also listed in this web application and the CarFitPlanner can assign a scheduled carwash to a CarFitWashers on a specific day and time. This carwash will be saved in a Gmail for the CarFitWashers and the CarFitWashers will check his calendar and carry out scheduled carwashes.

Future situation
(and why CarFit wants changes)

CarFit wants an app that displays the scheduled carwashes inside the app instead of a Gmail calendar. With this app, they can get feedback on where the CarFitWashers are located and how much time is required on a carwash. It is also possible to send push notifications so the CarFitWashers can navigate easily.

In this test, you shall make one screen in this new app CarFit wants to build.


YOUR
TEST

What you should do in this test and what is already done.

We have created an iOS solution, set up with the MVVM Architecture in mind. You have to create required model(s) ,view-model(s), parser and any other extensions. Inside the views folder in this solution there is one Main.storyboard and two subfolders that includes calendar UI and the home screen cell UI. So the UI required for this test has already been developed. This UI is made by a frontend developer without the knowledge of binding the UI components to the viewmodels and without any functionality.

Your job is to develop a proper functionality for the calendar that works as expected. Next, you need to bind all the UI components to the ViewModel and datasource and code the functionalities as per the business logic of that page.

You shall not do any UI in this test but use existing UI componets provided. If you find any errors in the Storyboard or xib you can fix it though, but it is not required, better to let us know when delivering your test.

Application description videos

These small videos will describe the functionality and data in the test. Videos will show an already implemented solution with the correct behavior. Data on the pages are in the Danish language, but you will understand seeing videos

(All 3 videos and the voice over is made by one of our Indian employees that is an “iOS UI developers” expert)

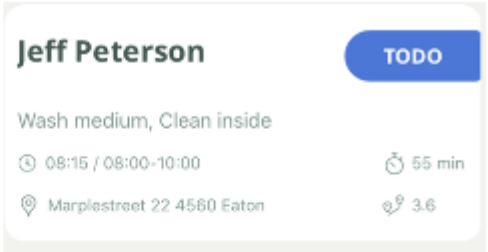
https://1drv.ms/v/s!Ap3qEzIs1WzjgkNo0O8KuRp0blqh?e=BUOFUq

https://1drv.ms/v/s!Ap3qEzIs1WzjgkUCgy0KnDAhGCZ9?e=0V4Le2

https://1drv.ms/v/s!Ap3qEzIs1WzjgkSxkWchxmmBT9mS?e=bWt31l

Data that will be used from json

CarwashVisit



- Jeff Peterson = houseOwnerFirstName + houseOwnerLastName
- TODO = visitState
- 08:15 = startTimeUtc (only the timepart)
- 08:00/10:00 = expectedTime
- Marplestreet 22 4560 Eaton = houseOwnerAddress + houseOwnerZip + houseOwnerCity
- 3.6 = houseOwnerLatittude + houseOwnerLongitude (used to calculate distance between two CarwashVisits)

Tasks

- Wash medium, Clean inside = title
- 55 min = timeInMinuttes (sum of task times)


```
    "tasks": [
        {
            "taskId": "e4131f30-6beb-45aa-9fcf-02758f332219",
            "title": "Pudsning indvendig",
            "isTemplate": false,
            "timesInMinutes": 25,
            "price": 99.00,
            "paymentTypeId": "51d14bdf-8d83-49fa-843d-787ceba4bb40",
            "createDateUtc": "2020-04-29T20:01:56.8831511",
            "lastUpdateDateUtc": "2020-04-29T20:01:56.8831539",
            "paymentTypes": null
        },
        {
            "taskId": "56ed0718-2854-4b11-8fde-8a3c573c9283",
            "title": "Pudsning udvendig",
            "isTemplate": false,
            "timesInMinutes": 35,
            "price": 199.00,
            "paymentTypeId": "9f40509b-191d-4c61-946e-0e816b63088d",
            "createDateUtc": "2020-04-29T20:01:26.2182451",
            "lastUpdateDateUtc": "2020-04-29T20:01:26.2182775",
            "paymentTypes": null
        }
    ],
```

```
"visitId": "799d7830-a5ff-4a0b-b571-9052df6a2e64",
"homeBobEmployeeId": "85f99bd7-ffd2-4ca7-1174-08d7984a4cf3",
"houseOwnerId": "4ad40a8b-8c55-45a3-9b81-b87527f5cf93",
"isBlocked": false,
"startTimeUtc": "2020-04-29T08:05:00",
"endTimeUtc": "2020-04-29T08:35:00",
"title": "Add title",
"isReviewed": false,
"isFirstVisit": false,
"isManual": false,
"visitTimeUsed": 0,
"rememberToday": null,
"houseOwnerFirstName": "Tone",
"houseOwnerLastName": "Holtermann",
"houseOwnerMobilePhone": "+4520934021",
"houseOwnerAddress": "Akademivej 15, 1 th",
"houseOwnerZip": "2800",
"houseOwnerCity": "Kgs. Lyngby",
"houseOwnerLatitude": 55.778830,
"houseOwnerLongitude": 12.521240,
"isSubscriber": false,
"rememberAlways": null,
"professional": "Test",
"visitState": "Done",
"stateOrder": 1,
"expectedTime": "08:00/10:00",
```

Functionality

- Use supplied JSON to generate UI of the app. The JSON supplied is only an example, not the full data source you may need, please generate more carwashes if needed. You might need to modify the dates so that you can test.
- We have already added the JSON in the test XCode project. You need to create a parser for that JSON and load the same into model class, if you can create a generic function that would be great.
- Create a CleanerListViewModel.swift that will have methods for fetching the data and display the same in UI. Follow MVVM design pattern. (The picture on the previous page will explain the relationship between JSON and data on the card.)
- To get the time for how long a visit will take (stopwatch icon) you have to sum up the time on each task.
- We want you to find a solution for getting the distance between to visits next to each other. It does not have to be driving distance coming from a map API. Just distance between two addresses latitude and longitude.
- We would like you to implement a refresh functionality, when you drag list down it will be refreshed. We know that you only have a static database that really does not refresh anything, but you can do the code anyway. The spinner should be visible when refreshing.
- The state a carwash can be in is Todo, InProgress, Done and Rejected. Each state will have its own color and is already included in the Colors.swift
    - Todo = #4E77D6
    - InProgress = #F5C709
    - Done = #25A87B
    - Rejected = #EF6565

Functionality (continued)

- We want the status colored label on a card in a list to change depending on the state of the carwash.
- When you click the calendar icon the calendar view should appear from the top. When you click anywhere on the page outside the calendar it should disappear with the simple animation.
- We need you to implement the calendar logic. When you click the left and right arrows calendar should show next month or last month. Also, the list with month days should be populated correctly, matching selected month. When you arrive on the present month it should be selected by default.
- It would be nice if it was possible to select two different days in the calendar of your choice and see the carwashlist being populated with those days. (Just give an example and tell us which days and month we can switch between). For example when 01 Jun is selected show data for that, similarly Jun 02, 03 etc.
- Other
    - Please do not use cocoapods, carthage or any third party libraries.
    - We only require that we can run your solution without doing any additional setup or third party services.

How will we judge your solution

Here is an overview of some of the most important parameters we will evaluate your test on

- We want to see code that uses good practices from OOP. Good use of SOLID principles, Protocols, Dependency Injection, MVVM, Naming convention, Clean code, comments wherever necessary explaining why you have used this approach.
- Your code can handle failures and recover from them. Exception Handling and code that use if/else to test that object, parameters are in a correct state, ex. not null.
- Your code works.

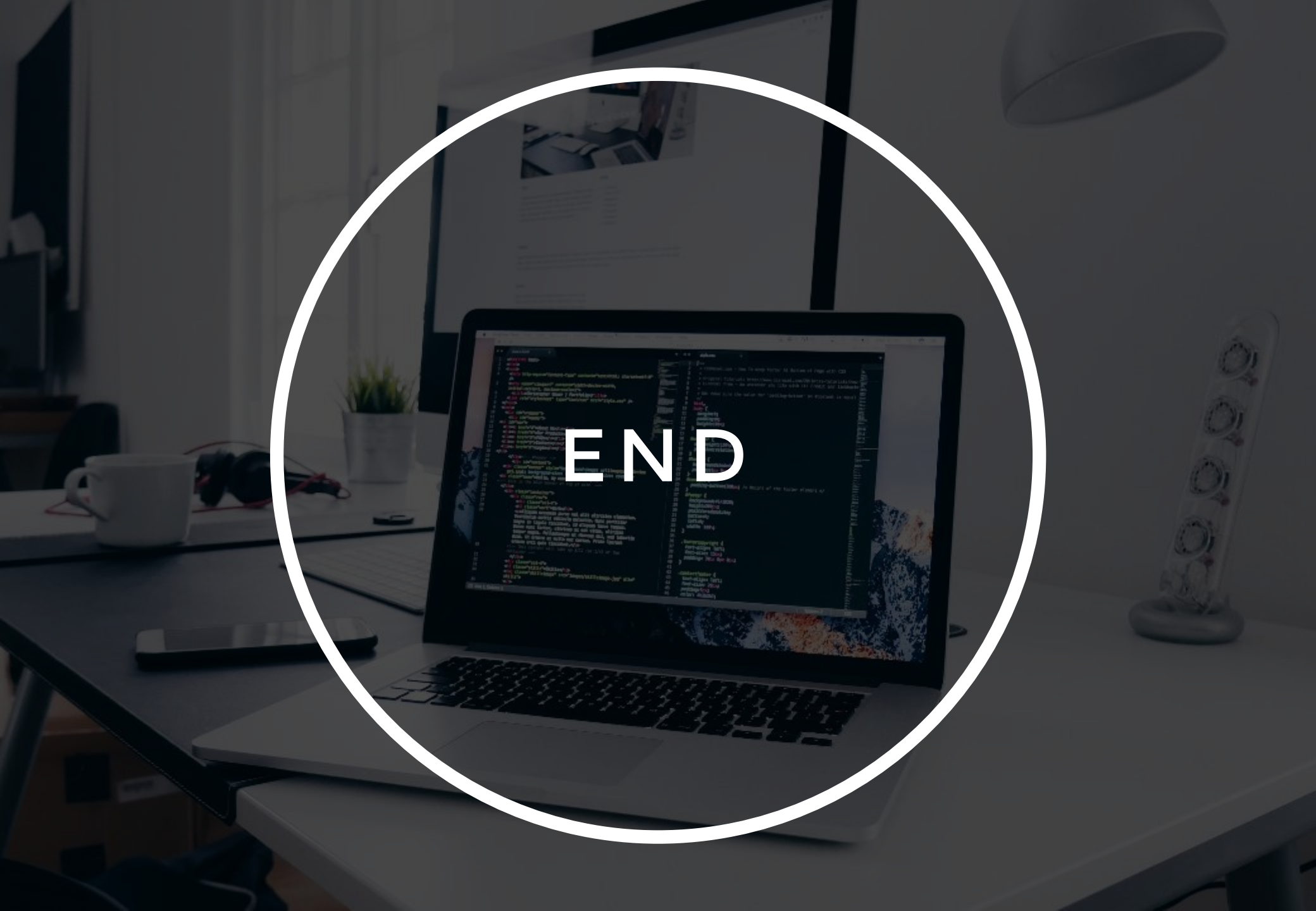
How will we NOT judge your solution

- We will not judge your test on the time you had used to make it. The most important is NOT that you make the test fast but that you make a good test with high quality. Time is not important for us at all. If we give you a job you will be working for us a fixed number of hours per week and will get the same payment whether you work fast or slow. We only care about quality. And you shall not tell us how long time you have used to make the test. So time will not be criteria at all.


YOUR TEST
DELIVERABLES

How to deliver your test

- Your solution and deliverables for this test consist of one (working) XCode project. Please make sure it will build in XCode. Also, remember to test your code in all iPhone devices available in the simulator.
    - If you would like to attach some prose description regarding your solution that is fine (this is completely voluntary and not a requirement). But you must only deliver documents (Word, PowerPoint and PDF), pictures, screenshots and videos.
- You shall gather all your files (XCode project, description documents, etc.) regarding this test in one folder and call this folder “iOS-UI-Data Binding-Developer-Test”. You shall NOT upload/send this folder directly to us on LinkedIn. Instead, upload the folder (with all the files) to WeTransfer (WeTransfer.com is free to use and requires no user) and upload the WeTransfer link to your test solution to me on LinkedIn.


END