Employee Monthly Pay Slip Generator

ReadMe

Chola Kallepalli

Developer  - srvarma7@gmail.com

Table of Contents

[What is the application for? 2](#_Toc71719151)

[How does the program work? 2](#_Toc71719152)

[Example input 2](#_Toc71719153)

[Example output 2](#_Toc71719154)

[Application demo 3](#_Toc71719155)

[Designing the solution 4](#_Toc71719156)

[Approach 4](#_Toc71719157)

[Ideology 4](#_Toc71719158)

[Design decision 4](#_Toc71719159)

[How to run the project 5](#_Toc71719160)

[Running the code 5](#_Toc71719161)

[Running the tests 5](#_Toc71719162)

[Assumptions 5](#_Toc71719163)

[Trade-offs 5](#_Toc71719164)

# What is the application for?

This program is a console application that allows to generate monthly pay slip for an employee based on their annual salary related to its tax rates.

# How does the program work?

After running the application using the Xcode IDE (Integrated development environment). The applications welcome’s the user and displays the instructions to follow for generating monthly payslip.

The user simply has to type their input to the console is a specific manner. For example, if a user wants to generate monthly payslip for Mary Song who makes 60000 a year as salary. So, the user has to enter – `GenerateMonthlyPayslip “Mary Song” 60000`

|  |
| --- |
| **Notice: -** As you can observe the first field **GenerateMonthlyPayslip** is the one we want to generate (hence calling it as keyword throughout the application). The second field within the double quotes is the **employee’s name**. And lastly, the field at the end is the **employee’s salary per annum**. |

The program checks if the entered data is valid or not. If valid, it calculates and displays the monthly payslip for Mary Song. If not valid, the application displays the error message based on the issue then asks user to re-enter the details.

Example input

|  |
| --- |
| GenerateMonthlyPayslip “Mary Song” 60000 |

Example output

|  |
| --- |
| \*\*\*\*\*\*\*\*\* Generated Monthly Payslip \*\*\*\*\*\*\*\*\*  -> Monthly Payslip for: "Mary Song"  -> Gross Monthly Income: $5000.0  -> Monthly Income Tax: $500.0  -> Net Monthly Income: $4500.0  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |

### Application demo

Text

Description automatically generated

The above is a sample demonstration when the input meets requirements.

A few examples of invalid data entry -

Text

Description automatically generated

The above is a sample demonstration when the input is empty (nothing is entered to console).

Text

Description automatically generatedText

Description automatically generated

The above are the sample demonstration when employee name entered incorrectly.

|  |
| --- |
| **Warning: -** When testing the application, it is essential to type the input in to the console rather than coping and pasting. If not done properly, the application is designed to force crash on purpose as we don’t want to show wrong output that is especially related to financials. |

# Development environment

Environment – Mac

OS – macOS Big Sur

Version - 11.3.1

IDE – Xcode

Language – Swift

Version – 5.4

# Designing the solution

## Approach

When finding a solution for the problem it is always important to first understand the requirement and think of the different possible ways to achieve the goal.

As always, I tried to understand the requirement before beginning to work for a solution. After understanding the problem and how the tax is being calculated for different salary slabs (ranges), started working in creating a very high-level flowchart diagram.

Diagram

Description automatically generated

Initial high-level flowchart

The program has been slightly modified during the development for improvements and based on the functionalities.

## Ideology

**Creating the skeleton of the application**

First created the abstract classes for the application using protocol to define the required functionalities. Using the defined protocol, we can make the class confirm to protocol for autogenerating the helper methods in the codebase.

**Distinguishing UI and program logic**

The application is designed such a way that a particular class does its specific things rather than just populating with functions or variables as needed. For instance, the Display class is responsible for displaying/outputting the response or feedback on the console. It is not advised to have a functionality to take/read input from console. This helps us to achieve encapsulation.

**Tax calculation**

**Error handling**

## Design decision

Protocols

MVVM

Encapsulation

# How to run the project

To run the project please follow the below steps.

Step 1:

## Running the code

## Running the tests

# Assumptions

Name of the employee

# Trade-offs

Integer as output