Idea of the project

The Loan Calculator In Python is a simple project developed using Python. The project is for the user convenient, to help them find out about the monthly payments on a specific loan. This project is for financial calculating to figure out the loan’s regular monthly payment, with total payment and the total interest. And the user has to pay those payments over the duration of the loan. This is a simple GUI-based project which is very easy to understand and use. Also, this project makes a convenient way for the user to gain an idea of how to perform loan calculations. Talking about the features of this system, this python application is designed to calculate monthly payments to expect for a simple loan. So, you can simply enter the loan amount, the term(years) and interest rate in the text fields and click ‘calculate loan’ button. It is not capable of handling all types of exceptions.

Algorithm

For GUI

1. Start
2. Create a heading label that display the calculator’s name
3. Create label and line edit pair for interest rate, label to show what user has to enter and line edit to enter text
4. Similarly create a pair for number of year and for amount
5. Create a push button to calculate
6. Create a label to show the calculated monthly payment
7. Create a label to show the calculated total amount
8. End

For Calculation

1. Make the line edit to accept only the number as input
2. Add action to the push button
3. Inside the push button action get the text of the line edits
4. Check if the line edit text is empty or zero then return so that function will not execute further
5. Convert the text value into integer
6. Calculate the monthly amount and set this value to the label
7. Calculate total amount from the monthly amount and show this value through label
8. End

Code

# importing required libraries

from PyQt5.QtWidgets import \*

from PyQt5 import QtCore, QtGui

from PyQt5.QtGui import \*

from PyQt5.QtCore import \*

import sys

class Window(QMainWindow):

    # constructor

    def \_\_init\_\_(self):

        super().\_\_init\_\_()

        # setting title

        self.setWindowTitle("Python ")

        # width of window

        self.w\_width = 400

        # height of window

        self.w\_height = 500

        # setting geometry

        self.setGeometry(100, 100, self.w\_width, self.w\_height)

        # calling method

        self.UiComponents()

        # showing all the widgets

        self.show()

    # method for adding components

    def UiComponents(self):

        # creating head label

        head = QLabel("Loan Calculator", self)

        # setting geometry to the head

        head.setGeometry(0, 10, 400, 60)

        # font

        font = QFont('Times', 15)

        font.setBold(True)

        font.setItalic(True)

        font.setUnderline(True)

        # setting font to the head

        head.setFont(font)

        # setting alignment of the head

        head.setAlignment(Qt.AlignCenter)

        # setting color effect to the head

        color = QGraphicsColorizeEffect(self)

        color.setColor(Qt.darkCyan)

        head.setGraphicsEffect(color)

        # creating a interest label

        i\_label = QLabel("Annual Interest", self)

        # setting properties to the interest label

        i\_label.setAlignment(Qt.AlignCenter)

        i\_label.setGeometry(20, 100, 170, 40)

        i\_label.setStyleSheet("QLabel"

                              "{"

                              "border : 2px solid black;"

                              "background : rgba(70, 70, 70, 35);"

                              "}")

        i\_label.setFont(QFont('Times', 9))

        # creating a QLineEdit object to get the interest

        self.rate = QLineEdit(self)

        # accepting only number as input

        onlyInt = QIntValidator()

        self.rate.setValidator(onlyInt)

        # setting properties to the rate line edit

        self.rate.setGeometry(200, 100, 180, 40)

        self.rate.setAlignment(Qt.AlignCenter)

        self.rate.setFont(QFont('Times', 9))

        # creating a number of years label

        n\_label = QLabel("Years ", self)

        # setting properties to the years label

        n\_label.setAlignment(Qt.AlignCenter)

        n\_label.setGeometry(20, 150, 170, 40)

        n\_label.setStyleSheet("QLabel"

                              "{"

                              "border : 2px solid black;"

                              "background : rgba(70, 70, 70, 35);"

                              "}")

        n\_label.setFont(QFont('Times', 9))

        # creating a QLineEdit object to get the years

        self.years = QLineEdit(self)

        # accepting only number as input

        onlyInt = QIntValidator()

        self.years.setValidator(onlyInt)

        # setting properties to the rate line edit

        self.years.setGeometry(200, 150, 180, 40)

        self.years.setAlignment(Qt.AlignCenter)

        self.years.setFont(QFont('Times', 9))

        # creating a loan amount label

        a\_label = QLabel("Amount", self)

        # setting properties to the amount label

        a\_label.setAlignment(Qt.AlignCenter)

        a\_label.setGeometry(20, 200, 170, 40)

        a\_label.setStyleSheet("QLabel"

                              "{"

                              "border : 2px solid black;"

                              "background : rgba(70, 70, 70, 35);"

                              "}")

        a\_label.setFont(QFont('Times', 9))

        # creating a QLineEdit object to get the amount

        self.amount = QLineEdit(self)

        # accepting only number as input

        onlyInt = QIntValidator()

        self.amount.setValidator(onlyInt)

        # setting properties to the rate line edit

        self.amount.setGeometry(200, 200, 180, 40)

        self.amount.setAlignment(Qt.AlignCenter)

        self.amount.setFont(QFont('Times', 9))

        # creating a push button

        calculate = QPushButton("Compute Payment", self)

        # setting geometry to the push button

        calculate.setGeometry(125, 270, 150, 40)

        # adding action to the calculate button

        calculate.clicked.connect(self.calculate\_action)

        # creating a label to show monthly payment

        self.m\_payment = QLabel(self)

        # setting properties to m payment label

        self.m\_payment.setAlignment(Qt.AlignCenter)

        self.m\_payment.setGeometry(50, 340, 300, 60)

        self.m\_payment.setStyleSheet("QLabel"

                                     "{"

                                     "border : 3px solid black;"

                                     "background : white;"

                                     "}")

        self.m\_payment.setFont(QFont('Arial', 11))

        # creating a label to show monthly payment

        self.y\_payment = QLabel(self)

        # setting properties to y payment label

        self.y\_payment.setAlignment(Qt.AlignCenter)

        self.y\_payment.setGeometry(50, 410, 300, 60)

        self.y\_payment.setStyleSheet("QLabel"

                                     "{"

                                     "border : 3px solid black;"

                                     "background : white;"

                                     "}")

        self.y\_payment.setFont(QFont('Arial', 11))

    # method for calculating monthly

    # and annually payments

    def calculate\_action(self):

        # getting annual interest rate

        annualInterestRate = self.rate.text()

        # if there is no number is entered

        if len(annualInterestRate) == 0 or annualInterestRate == '0':

            return

        # getting number of years

        numberOfYears = self.years.text()

        # if there is no number is entered

        if len(numberOfYears) == 0 or numberOfYears == '0':

            return

        # getting loan amount

        loanAmount = self.amount.text()

        # if there is no number is entered

        if len(loanAmount) == 0 or loanAmount == '0':

            return

        # converting text to int

        annualInterestRate = int(annualInterestRate)

        numberOfYears = int(numberOfYears)

        loanAmount = int(loanAmount)

        # getting monthly interest rate

        monthlyInterestRate = annualInterestRate / 1200

        # calculating monthly payemnt

        monthlyPayment = loanAmount \* monthlyInterestRate / (1 - 1 / (1 + monthlyInterestRate) \*\* (numberOfYears \* 12))

        # setting formatting

        monthlyPayment = "{:.2f}".format(monthlyPayment)

        # setting text to the label

        self.m\_payment.setText("Monthly Payment : " + str(monthlyPayment))

        # getting total payment

        totalPayment = float(monthlyPayment) \* 12 \* numberOfYears

        totalPayment = "{:.2f}".format(totalPayment)

        # setting text to the label

        self.y\_payment.setText("Total Payment : " + str(totalPayment))

# create pyqt5 app

App = QApplication(sys.argv)

# create the instance of our Window

window = Window()

# start the app

sys.exit(App.exec())

Modules used

1. **QtCore**

Core non-GUI classes used by other modules

1. **QtGui**

Graphical user interface components

1. **QtWidgets**

Classes for creating classic desktop-style UIs

Implementation

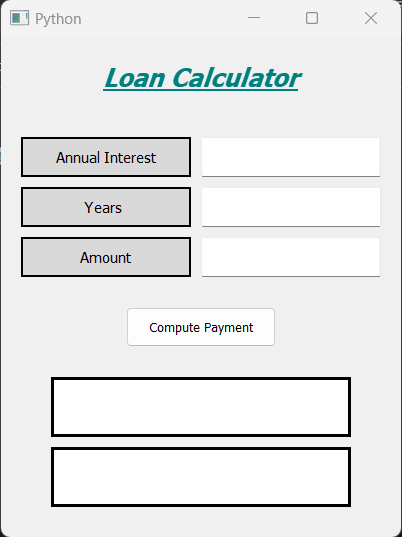
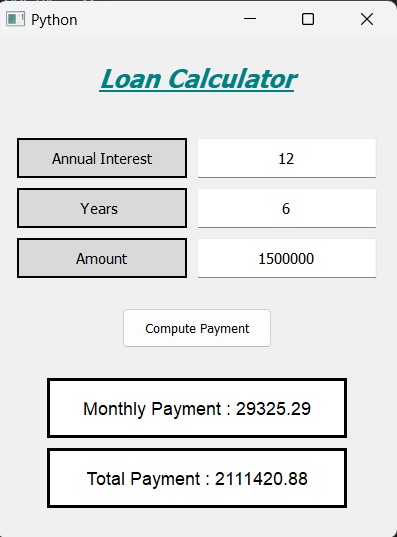
The task is for the client advantageous, to assist them with getting some answers concerning the regularly scheduled instalments on a particular credit.

This venture is for monetary ascertaining to make sense of the credit’s customary regularly scheduled instalment, with all out instalment and the absolute intrigue. Also, the client needs to pay those instalments over the span of the advance.

Discussing the highlights of this loan calculator framework, this python application is intended to compute regularly scheduled instalments to expect for a straightforward credit.

Thus, you can basically enter the credit sum, the term(years) and financing cost in the content fields and snap ‘ascertain advance’ button. It isn’t equipped for dealing with a wide range of exemptions.

Output

Conclusion

In this project, we developed a basic loan calculator using some basic functions and PyQt5 tool kit / GUI tool kit. We made a GUI (graphic user interface). In this loan calculator, we that user can input his required sum of money, interest percentage and years. The calculator gives out the monthly payment as well as total payable amount.

Sources

<https://www.geeksforgeeks.org/>

<https://www.youtube.com/>

<https://pypi.org/project/PyQt5/>

Introduction

You will probably need to take out a loan during your lifetime. Whether you’re buying a car or a home, starting a business, consolidating debt, facing unexpected expenses, or paying for a college education, there will be situations when you’ll need more money for a purchase or a bill than you have in savings.

A loan is often a better choice for a ready source of cash than a credit card because interest rates on loans are generally considerably lower than on purchases that you make via credit cards. There are some exceptions to that rule—[payday loans](https://www.investopedia.com/terms/p/payday-loans.asp), for one—but using a loan calculator will help you determine whether the loan that you’re considering is fairly priced and right for you.

Use this calculator to test out any loan that you are considering. By tweaking the loan amount, loan term, and interest rate, you can get a sense of the possible overall cost. You will see that as the term of the loan increases, your monthly payments go down, but the overall cost of the loan (the total paid) rises. Be sure to consider any fees that you may be charged for mortgages and other specific types of loans.You can also look at the loan calculator in reverse. Find out how much you can afford to borrow based on a monthly payment that you can afford at the interest rates that you may be offered. Even if a bank or a car dealership is offering to lend you the money, it’s your job to determine whether that would be a wise financial move for you.

**Secured vs. Unsecured Loans**

Mortgages and car loans are offered at lower interest rates than personal loans because they are secured by the collateral of the house or car from which you are borrowing money to buy.

A personal loan with no collateral against it will cost you more in interest because if you default, then the bank will have nothing tangible to foreclose on or repossess to cover your debt. These are known as unsecured loans, and typically carry higher interest rates, all else equal.

**How Much Loan Can I Afford?**

You should be able to adequately cover the monthly mortgage payments, including principal, interest, and any fees with the income you currently generate. In general, it is recommended to spend no more than 30% of your gross (pre-tax) income on a mortgage and 10%-15% on an auto loan.

