CÓDIGO ADWARE:

#!/usr/bin/env python3

""" Implementation of simple adware that pops multiple

windows with the advertisements.

"""

import logging

import sys

import random

from PySide2.QtWidgets import QApplication, QDialog, QLabel, QVBoxLayout

class AdWindow(QDialog):

    """ This class represents ad window shown on the screen. """

    def \_\_init\_\_(self, ad\_slogan, parent=None):

        super(AdWindow, self).\_\_init\_\_(parent)

        self.setWindowTitle("Advertisement!")

        # Create a layout so that the ad slogan is shown.

        self.label = QLabel(ad\_slogan)

        layout = QVBoxLayout()

        layout.addWidget(self.label)

        self.setLayout(layout)

    def closeEvent(self, event):

        # Ignore the close event so that the ad

        # can't be closed by pressing close button.

        event.ignore()

class Adware(QApplication):

    """ This class represents implementation of adware. """

    def \_\_init\_\_(self, args):

        super(Adware, self).\_\_init\_\_(args)

    @property

    def advert\_slogans(self):

        """ Slogans of the promoted adds. """

        return (

            'Buy the milk in the milk shops!',

            'Buy the clothes in the wool shops!',

            'Buy the food in the food shops!'

        )

    def create\_ad\_window(self, ad\_slogan):

        """ Creates a windows showing the advertisement

        slogan.

        :param str ad\_slogan: Text of the ad.

        """

        window = AdWindow(ad\_slogan=ad\_slogan)

        window.show()

        return window

    def show\_ads(self):

        """ Creates the main GUI application and shows

        the ads based on `:class:~Adware.advert\_slogans`

        """

        ad\_windows = []

        for advert in self.advert\_slogans:

            # Create a new ad window.

            ad\_window = self.create\_ad\_window(advert)

            # Move this window to random location on screen.

            x\_coordinate, y\_coordinate = random.randint(1, 800), random.randint(1, 600)

            ad\_window.move(x\_coordinate, y\_coordinate)

            ad\_windows.append(ad\_window)

        return ad\_windows

if \_\_name\_\_ == '\_\_main\_\_':

    logging.basicConfig(level=logging.DEBUG)

    # Create our adware and show the ads.

    adware = Adware(sys.argv)

    windows = adware.show\_ads()

    sys.exit(adware.exec\_())

**How does it work**

* Firstly, we create our **adware** and pass it arguments from the system. Because we need a proper GUI, we are using Python module called [PySide2](https://pypi.org/project/PySide2/). To learn more about GUI programming, see [the guide to GUI programming in Python](https://wiki.qt.io/Qt_for_Python). Our class **Adware** inherits from the **QApplication** and represents main QT application.

adware = Adware(sys.argv)

* We call the method *show\_ads()*, which creates dialog popups and pass the references to these forms to variable in main module windows. It is important not to loose reference to those windows, because otherwise they will not be shown on the screen.

windows = adware.show\_ads()

* Our adware has a property **advert\_slogans**, which represents a list of ad slogans we want our victim to see. For each of those slogans we want to create a unique popup window by calling the method *create\_ad\_window()*.
* ad\_windows = []
* for advert in self.advert\_slogans:
* # Create a new ad window.

ad\_window = self.create\_ad\_window(advert)

* Because these windows would popup on the same place on the screen and overlap each other, we need to move the created popup windows to random location on the screen.
* # Move this window to random location on screen.
* x\_coordinate, y\_coordinate = random.randint(1, 800), random.randint(1, 600)

ad\_window.move(x\_coordinate, y\_coordinate)

* To create a popup windows, our function *create\_ad\_window* creates a new **AdWindow** with the given slogan. To show the window on the screen, we must call the method *show*.
* window = AdWindow(ad\_slogan=ad\_slogan)

window.show()

* The popup window called **AdWindow** inherits from **QDialog** and represents independent window with layout containing only one label showing the ad. However, to make adware more annoying and show ads more aggressively, we set the window to ignore close signal when the victim presses close button. When this happens, the window obtaines information about a new event called **closeEvent**. We will simply ignore any action so the window stays on the screen.
* def closeEvent(self, event):

event.ignore()