Objectgeoriënteerd Programmeren: WPO 6 (bestanden)

1. Inhoud

Tekstbestanden (lezen & schrijven), try-catch, IOException, CSV-formaat, Notepad++

2. Opgaves

- Demo 1: Tabel in CSV-formaat, schrijven en lezen
- A: Notepad
- A: Config file
- A: FileDialog
- A: Menu's
- A: Mouse follower
- A: Commentaar

Hint: Vermits we met bestanden zullen werken, is het gebruik van een externe teksteditor een bijna noodzakelijke hulp. Een goede editor hiertoe is "Notepad++". Met dit programma kunnen bijna alle type tekstbestanden geanalyseerd worden! Gebruik dit programma dan ook om de bestanden te verifiëren! Dit programma is te verkiezen boven de klassieke "Notepad" van Microsoft. Kijk daarom al je bestanden na tijdens het programmeren. In deze les behandelen we enkel bestanden die tekst bevatten (dus geen binaire informatie).
Nuttige link: https://msdn.microsoft.com/en-us/library/system.io.file.appendtext%28v=vs.110%29.aspx

2.1 Demo 1: Tabel in CSV-formaat, schrijven en lezen

Maak een programma die het CSV-bestand (zie website) kan openen, lezen en de inhoud ervan kan weergeven (kies zelf de wijze van weergave). Laat de gebruiker toe om data in te voeren (enkel waarden relevant voor de kolommen), en dit in hetzelfde formaat achteraan het bestand toe te voegen. Let op dat alle waarden gescheiden blijven door komma's, en dat de kommagetallen voorgesteld worden door een punt. Vervang hiervoor ("string replace") de ',' door een '.' vlak voor het wegschrijven naar het bestand.

2.2 A: Notepad

Schrijf een programma waarin je de basisfuncties van notepad overneemt: een tekstbestand openen en weergeven in een grote tekstbox. Eens de gebruiker gedaan heeft met het schrijven van tekst in de tesktbox, wordt de data opgeslagen in hetzelfde bestand. In deze opgave mag je als bestandsnaam een vaste string programmeren. Kan je ook een bestand aanmaken indien het programma eerst geen bestand heeft ingelezen?

2.3 A: FileDialog

Herneem de voorgaande opgave (Notepad) van deze les. Vervang de constante string die je hanteerde als bestandsnaam door het oproepen van een FileDialog (SaveFileDialog, OpenFileDialog). Roep het dialoogvenster aan, en ga na op welke wijze je een bestandsnaam uit het dialoogvenster kan extraheren indien men op "OK" geklikt heeft. Deze bestandsnaam gebruik je om bestanden te openen, ernaar te schrijven of er uit te lezen. Zoek op hoe je een dialoogvenster oproept?

2.4 A: Menu's

Herneem de voorgaande opgave. Vervang de knoppen op het formulier (om uit een bestand te lezen of er naar een bestand op te slaan) door een menubalk (zie Toolbox Visual Studio). Experimenteer met de menubalk (submenu's, enz.) en voeg de nodige handelingen hierin toe. Het oproepen van een event-handler gebeurt op dezelfde wijze als voor een gewone button.

2.5 A: Config file

Schrijf een programma dat het bestand "LED.conf" (zie website) kan openen, lezen en de inhoud ervan kan weergeven. Dit bestand bevat de configuratie van 5 opeenvolgende LED's (RGB of kleuren). Teken 5 LED's (met een diameter van 25 pixels) naast elkaar in een canvas (100 hoog, 300 breed) en zorg ervoor dat ze de juiste kleurencombinatie krijgen zoals aangeduid in het bestand. Je mag ervan uitgaan dat LED 1 de meest links getekende LED is en LED 5 (in de configuratie) de meest rechts getekende LED is. Kijk het bestand eerst na met Notepad++.

2.6 A: Mouse follower

In deze opdracht schrijf je een programma die toelaat om de positie van de muiscursor op een canvas te loggen in een CSV-bestand. Je start hierbij van de aangeleverde solution. Zorg ervoor dat je de positie (x,y) en het tijdstip van deze positie opslaat in een CSV-bestand. Hierbij neem je kolom 1 voor de tijd (in ms), kolom 2 voor de x-positie en kolom 3 voor de y-positie. Ga na dat je bestand correct is.

Voorzie een knop "Lees bestand" waarin je het bestand gaat uitlezen (al dan niet met een dialoogvenster). Daarin teken je het pad dat de cursor heeft gevolgd in een canvas.

2.7 A: Commentaar

Sommige bestanden (zoals bestanden met programmeercode) kunnen regels bevatten waarin commentaar aanwezig is. Commentaar kan nuttig zijn in het geval dat men de volgende regels van het bestand wilt uitleggen, of om een setting uit te schakelen. Een veelgebruikt teken als commentaar is het symbool: "#". Door dit symbool als eerste teken binnen een regel te schrijven, wordt de rest van de regel gewoonweg genegeerd. Nagaan of dat het eerste teken van de regel het commentaarsymbool bevat, kan gebeuren door het eerste teken van string te vergelijken op "#" (bv. str[0]==‘#’).

Opgave: Schrijf een programma dat het bestand "commentaar.txt" kan inlezen. Alle regels worden in een tekstbox weergegeven. Regels beginnende met het commentaarsymbool worden hierbij niet in de textbox weergegeven.

Extra: Soms kan het gebeuren dat het commentaar symbool niet het eerste symbool in de regel is, maar pas na een aantal tekens voorkomt. Zorg ervoor dat alles wat voor het commentaarsymbool komt, weergegeven wordt. Alles erna (op de regel) wordt genegeerd.

Hint: In sommige bestanden worden spaties tussen waarden gezet. Dit verhoogt de leesbaarheid voor het menselijk oog. Voor configuratiebestanden is dit echter helemaal niet nodig, en kan tijdens het lezen juist lastig worden voor het programmatuur (vergelijken van tekenreeksen). Met de functie "trim" haal je overbodige spaties (voor de tekenreeks als na de tekenreeks) weg. Pas dit eventueel toe in andere projecten.

Hint: Het vergelijken van tekenreeksen gebeurt a.d.h.v. een vergelijkingsreeks (tekenreeks). Vaak wordt er maar één variant voorzien. Om ervoor te zorgen dat hoofd- en kleine letters op dezelfde manier geïnterpreteerd worden, wordt de input vaak naar ofwel hoofdletters omgezet, ofwel naar kleine letters (to upper, to lower). Zo moet je maar 1 keer vergelijken, en niet alle mogelijkheden afgaan. Pas dit eventueel toe in andere projecten.