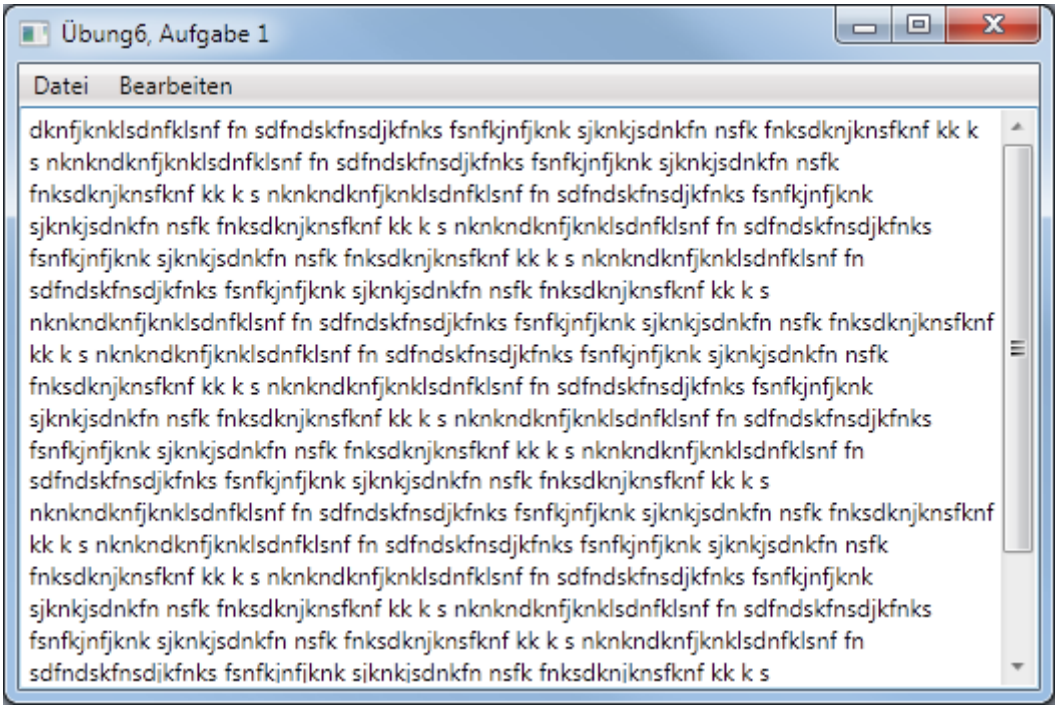
Windowsprogrammierung – WiSe17

Übung 7

Aufgabe 7.1 Texteditor

Schreiben Sie mit WPF einen einfachen Texteditor. Die graphische Oberfläche soll wie folgt aussehen:



Sie soll eine Menüleiste (Klasse *Menu*) enthalten, die stets an der oberen Kante des Clientbereiches angedockt ist und deren gesamte Breite umfasst, sowie eine *TextBox*, die den gesamten Rest des Clientbereiches einnimmt.
Die *TextBox* soll Zeilenumbrüche akzeptieren, Zeilen automatisch umbrechen und eine vertikale Scrollleiste anzeigen, wenn der Text länger als die Höhe der *TextBox* ist.
Das Menü soll folgende Unterpunkte bereitstellen:

- Datei

  - Speichern…
  - Laden…
  - Beenden
- Bearbeiten
  - Alles auswählen
  - Kopieren
  - Ausschneiden
  - Einfügen
  - Suchen…:

Bei den Menüpunkten „Speichern…“ und „Laden…“ soll jeweils ein Dialogfenster angezeigt werden, in dem die betreffende Datei ausgewählt werden kann. Es sollen nur Dateien im .txt-Format gespeichert und geladen werden können. Verwenden Sie für die Dialogfenster die Klassen *SaveFileDialog* und *OpenFileDialog*. Verwenden Sie die Klassen *StreamWriter* und *StreamReader* zum Schreiben in eine bzw. zum Lesen aus einer Textdatei.

Die Menüpunkte „Alles auswählen“, „Kopieren“, „Ausschneiden“ und „Einfügen“ können Sie elegant und einfach mit Hilfe von gerouteten WPF-Commands realisieren.

Bei dem Menüpunkt „Suchen…“ soll sich ein selbstgeschriebener Modaldialog öffnen, in dem der Benutzer einen Suchbegriff in eine *TextBox* eingeben kann und diesen mit Hilfe eines *Button*s bestätigen kann. Nach dem Klick auf den *Button* soll sich das Dialogfenster wieder schließen und es soll das erste Vorkommen des Suchbegriffs innerhalb des Textes, falls vorhanden, als markiert dargestellt werden. Definieren Sie innerhalb Ihrer Klasse für diesen Dialog eine lesbare Eigenschaft *Suchbegriff*, auf die das Hauptfenster nach dem Anzeigen des Dialoges zugreifen kann. Die eigentliche Suchen-Funktion soll nur dann ausgeführt werden, wenn der Benutzer den Dialog durch einen *Button*-Klick und nicht z.B. durch einen Klick auf das Schließen-Kreuz beendet.

Aufgabe 7.2 Der ängstliche Button

Schreiben Sie ein WPF-Programm mit den folgenden Eigenschaften:

Die Benutzeroberfläche soll aus einem *Canvas*-Panel bestehen, das den gesamten Clientbereich einnimmt. Das *Canvas* soll die folgenden 2 Kindelemente besitzen:

- Einen *Button* mit dem Inhalt „Versuch’s doch“, der sich beim Programmstart im *Canvas* an der Position (50,50) befindet. Die Breite und Höhe des *Buttons* soll sich beim Programmstart an seinem Inhalt orientieren.
- Einen *Slider* mit einer Breite von 50, der sich im *Canvas* an der Position (10,10) befindet.

Das Fenster soll eine Breite von 525 und eine Höhe von 350 haben. Sorgen Sie dafür, dass der Benutzer die Größe des Fensters nicht verändern kann.

Mit Hilfe des *Sliders* soll der Benutzer die Größe des *Buttons* ändern können. Dabei sollen folgende Regeln gelten:

- Die Größe des *Buttons* soll sich proportional zu seiner Anfangsgröße ändern. Damit ist gemeint, dass Breite und Höhe des *Buttons* stets ein konstantes Verhältnis zueinander haben, wobei dieses Verhältnis durch die Anfangsgröße vorgegeben wird.
- Der minimale Skalierungsfaktor für die Größe beträgt 1, der maximale Skalierungsfaktor für die Größe beträgt 5.
- Die Position der linken, oberen Ecke des *Buttons* soll von einer Größenänderung unbeeinflusst bleiben.
- Alle 4 Ecken des *Buttons* sollen stets sichtbar sein. Ist dies nach einer theoretischen Größenänderung nicht der Fall, soll keine durchgeführt werden.

Wird der Mauszeiger über den *Button* bewegt, soll dieser an eine zufällige andere Position „springen“. Für die Bestimmung der neuen Position des *Buttons* sollen folgende drei Einschränkungen gelten:

1. Alle 4 Ecken des *Buttons* sollen stets sichtbar sein.
2. Der Mauszeiger darf sich nach dem Sprung nicht innerhalb des *Buttons* befinden. Insbesondere soll es nicht möglich sein, den *Button* anzuklicken.
3. Der *Button* darf keinen Überschneidungsbereich mit dem *Slider* haben.

<u>Hinweise zur Bestimmung der neuen *Button*-Position:</u>

- Verwenden Sie eine Instanz der Klasse *Random* und deren Methode *NextDouble*, um eine Pseudozufallszahl im Intervall [0,1] zu erzeugen.
- Benutzen Sie die *Rect*-Struktur um eine Überschneidung von *Button* und *Slider* zu überprüfen.
- Ermitteln Sie zunächst eine zufällige Position, sodass zumindest die 1. Bedingung („Alle 4 Ecken des *Buttons*…“) erfüllt ist. Anschließend überprüfen Sie, ob für die ermittelte Position auch die 2. und 3. Bedingung gelten. Ist dies nicht der Fall, fangen Sie wieder von vorne an.