Entwicklung interaktiver Systeme

M. Sc. Tobias Puderer – WS 22/23

Übung 1: Installation Visual Studio und Einarbeitung

1. Installieren Sie Visual Studio Community Edition. Diese kann für Windows PCs kostenlos unter folgender URL heruntergeladen werden.

   https://www.visualstudio.com/products/free-developer-offers-vs.aspx

   Falls Ihnen kein Rechner mit Windows zur Verfügung steht: Im PC Pool, in welchen die Übung stattfindet, ist ebenfalls Visual Studio vorinstalliert.

   Machen Sie sich im Vorfeld bereits ein wenig mit WPF vertraut. In der Bibliothek finden Sie als ergänzende Literatur das Buch „Windows Presentation Foundation Unleashed“ von Adam Nathan. Zudem gibt es ein deutsches EBook „Visual C# 2012“, welches zwar zu Teilen etwas veraltet ist, aber dennoch einen guten Einblick in das Thema WPF liefert.

   http://openbook.rheinwerk-verlag.de/visual_csharp_2012/1997_18_001.html#dodtp34972f67-6c1a-406f-a91d-ec8ff67058f9

2. Kurzeinführung XML:

Der Begriff XML (Extensible Markup Language) beschreibt ein Format, in welchem Daten strukturiert und hierarchisch erfasst werden können. Es basiert, wie HTML, auf der Auszeichnungssprache SGML (Standard Generalized Markup Language) und besitzt eine weite Verbreitung in vielen Bereichen der Softwaretechnik. Im Gegensatz zu HTML existiert jedoch keine festgelegte Menge an möglichen Anweisungen. Vielmehr erlaubt XML durch den Einsatz von sog. Schemata oder DTDs (Dokumenttyp-Definition), die Regeln des Dokumentaufbaus selbst zu definieren.

Kenntnisse in XML zählen in der heutigen Zeit quasi zum Allgemeinwissen eines Softwareentwicklers, da es häufig bei der Strukturierung und Verarbeitung von Daten eingesetzt wird z.B. als zugrundeliegendes Format in Konfigurationsdateien oder im Kontext von Schnittstellenkommunikation.

XML Dokumente bestehen aus sog. *Entitäten* oder *Tags* (Einheiten), welche mit eckigen Klammern (< und >) umschlossen sind und durch Attribute erweitert werden können (Wenn Sie HTML kennen: Es ist das gleiche Schema, da HTML auf XML basiert). Jede dieser Entitäten kann weitere Einheiten enthalten, wodurch die angesprochenen hierarchischen Strukturen abgebildet werden können.

Nachfolgend ein Beispiel für eine mögliche Strukturierung der vorliegenden Übung im XML Format.

```
<?xml version ="1.0" ?>
<eis_uebungen>

  <!-- Ich bin ein Kommentar :) -->
  <uebung nr="1" name="Installation Visual Studio und Einarbeitung" >
    <beschreibung>
      Installieren Sie Visual Studio Community Edition. Dieses kann für ...
    </beschreibung>
  </uebung>

  <uebung nr="2" name="Erstellung und Aufbau eines WPF Projekts" >
     <aufgabenteil nr="1">
         <beschreibung>
          Öffnen Sie Visual Studio und erstellen Sie ein neues WPF Projekt.
         </beschreibung>
            ...
     </aufgabenteil>
      ...
   </uebung>

</eis_uebungen>
```

Da die Benamung der Tags frei wählbar ist, könnten Tags auf unterschiedlichen Ebenen gleich benannt werden. Dies kann die Lesbarkeit, aber auch die Möglichkeit zur gezielten Ansteuerung von Informationen durch Software negativ beeinflussen. Daher existiert zusätzlich die Möglichkeit, sog. Namensräume zu definieren, um die Bedeutung der Tags eindeutig zu halten.

```
<?xml version ="1.0"
  xmlns:p="http://www.beliebigeURL.hauptsache-eindeutig.de/person"
  xmlns:f="http://www.beliebigeURL.hauptsache-eindeutig.de/film" ?>
<info>
   <hauptdarsteller>
     <f:name>Fear and loathing in Las Vegas</f:name>
     <p:name>Johnny Depp</p:name>
   </hauptdarsteller>
</info>
```

Das Beispiel soll verdeutlichen, dass die beiden „name“ Tags nichts mit einander zu tun haben, da sie sich in unterschiedlichen Namensräumen befinden.

Um XML Dokumente verarbeiten zu können, müssen sie wohlgeformt und valide sein.

1. Wohlgeformte Dokumente erfüllen die syntaktischen Regeln der allgemeinen XML-Spezifikation. Hierunter fallen z.B. das Vorhandensein mindestens einer Entität und das korrekte Setzen der Klammerungen (< und >). Hat eine Entität keine weiteren Unter-Entitäten, so muss sie nicht mit Start- und Endtag umschlossen werden. Es genügt, an das Ende des Tags einen Slash zu setzen um dessen Vollständigkeit zu signalisieren.

   ```
   <!-- Ohne verschachtelte Elemente ist das ist nicht nötig -->
   <info attribut="wert">

   </info>

   <!-- Das tut‘s auch -->
   <info attribut="wert" />
   ```

2. Valide Dokumente sind wohlgeformt und erfüllen die strukturellen Regeln, die ihnen durch die Definition eines Schemas oder einer DTD[1] auferlegt wurden. Diese sind optional, eignen sich jedoch um sicherzustellen, dass ein XML-Dokument alle geforderten Daten enthält. Dies ist insbesondere bei der Veröffentlichung von

[1] DTDs und Schemas sind **nicht** Bestandteil der Übung. Es schadet allerdings auch nicht zu wissen, was damit gemeint ist. Wer sich dafür interessiert, kann sich z.B. an folgender Stelle weiterbilden: http://openbook.rheinwerk-verlag.de/kit/itkomp15001.htm

Schnittstellen wie z.B. Webservices hilfreich, die von anderen Anwendungen genutzt werden sollen.

Aufgabe: Lesen Sie zunächst die Kurzeinführung um sich einen ersten Eindruck von der strukturierten Beschreibungssprache XML zu machen. Erstellen Sie dann ein valides und wohlgeformtes XML Dokument, in welchem Sie folgende Daten strukturiert darstellen.

- Vorname
- Name
- Matrikelnummer
- Belegte Module im aktuellen Semester

Überlegen Sie sich hierfür eine passende Struktur. Orientieren Sie sich ggf. an den genannten Beispielen.