Definition des Dateiformats vom Tabellenauswerter

Version: 0.2 (Workaround für Sonderzeichen hinzugefügt)

Allgemeines
Die Dateiendung wird „.table” lauten.

Anforderungen
Beim Entwurf und Implementierung des Dateiformats muss folgendes Beachtet werden:

- Es muss erweiterbar sein, weil damit gerechnet wird, dass das Programm nach Projektende weiterentwickelt wird und dass die gesammelten Daten auch nach Jahren für den Kunden wichtig sein werden. Deswegen muss es möglich sein, dem Format später neue Elemente hinzuzufügen, ohne, dass die alten Dateien inkompatibel mit der neuen Programmversion sein würden.
- Es muss mit beliebigen Datenmengen funktionieren, weil diese Wahrscheinlich sehr groß werden. Pro Saison wird mit über 10.000 Teilnehmern gerechnet und es ist nicht bekannt, wie groß die Schnittmenge zwischen den Saisons ist. Der Kunde möchte auch die Teilnehmer speichern, die nur in einigen (bzw. auch eine) Saison(s) mitmachen. Deswegen muss bei der Implementierung darauf geachtet werden, dass die Größe der Datei, bzw. der enthaltenen Strukturen nicht durch irgendwelche Faktoren begrenzt wird.
- Sonderzeichen aus diversen Sprachen sollen kein zu großes Problem darstellen.

Struktur
Wie bereits erwähnt, wird die Datei genau eine Tabelle enthalten. Dabei wird jede Zeile einem Datensatz entsprechen. Jeder Datensatz besteht wiederum aus „Einträgen“.
Jeder Eintrag soll folgende Informationen enthalten:

- Die Daten, die vom Programm angezeigt werden. Für ihre Sortierung ist es sinnvoll verschiedene Typen von Daten zu unterscheiden. Die Spalten, die Zahlen enthalten sollen nämlich nach ihrem Wert sortiert werden und nicht alphabetisch.
    - Dafür soll es folgende Typen geben:
        - Reine Zeichenketten: Diese werden alphabetisch sortiert.
        - Reine Zahlen: Diese werden nach ihrem Wert sortiert.
        - Kombination aus Zeichenkette und Zahl (z.B. „50%“, „1. (-)“): Diese werden auch nach ihrem Wert sortiert.
- Optionaler Verweis: In den Webseiten, die das Programm ausliest, enthalten viele Einträge Verweise zu anderen Webseiten. Die enthaltene Adresse soll auch gespeichert werden.

Implementierung
Die Syntax orientiert sich an XML. Es gibt aber kein formales Schema, o.ä., weil es z.Z. nicht als nötig erachtet wird.
Die Datei baut sich aus folgenden Elementen auf:

`<tablefile version=”x.x”>...</tablefile>`
Umschließt die komplette Datei. Der Parameter „version” gibt die Version der Datei an. Damit wird versucht, sicher zu stellen, dass bei Änderungen im Dateiformat die alten Dateien weiterhin vom Programm eingelesen werden können.
Für Testzwecke lässt sich die Versionsnummer auf „0.0” setzen. Diese Nummer wird immer wie die aktuelle Version behandelt.
Unterhalb dieses Elementes muss ein „table”-Element vorkommen.

`<table>...</table>` (Element von „tablefile”)
Umschließt die eigentliche Tabelle. Unterhalb dieses Elementes kann als erstes ein „headerrow”-

Element vorkommen (muss aber nicht) und es können beliebig viele „row”-Elemente vorkommen (mindestens einer).

<headerrow>...</headerrow> (Element von „table”; optional)
Umschließt die Kopfzeile. In einer korrekten Datei kann dieses Element höchstens einmal, ganz oben vorkommen.
Unterhalb dieses Elementes können beliebig viele „entry”-Elemente vorkommen (mindestens einer).

<row>...</row> (Element von „table”)
Umschließt eine Zeile. Unterhalb dieses Elementes können beliebig viele „entry”-Elemente vorkommen (mindestens einer).

<entry>...</entry> (Element von „headerrow” und „row”)
Umschließt eine Zelle, bzw. Eintrag. Unterhalb dieses Elements muss ein „data”-Element vorkommen. Optional kann es noch ein „link”-Element enthalten.

<data>...</data> (Element von „entry”)
Umschließt den Bereich mit den eigentlichen Daten. Unterhalb dieses Elements muss ein „current”-Element vorkommen. Optional kann es noch ein „old”-Element enthalten.

<link addr=”*Adresse zu Webseite***”/>** (Element von „entry”; optional)
Enthält in dem Parameter die Adresse zu einer Webseite.

<current>...</current> (Element von „data”)
Enthält die aktuellen (bzw. zusammengerechneten) Daten. Dieses Element kann entweder einen „number”-Element, einen „string” Element oder beides enthalten. Wenn beides vorkommt, wird, unabhängig von der Reihenfolge, in der es in der Datei steht, davon ausgegangen, dass die die Nummer vor der Zeichenkette kommt.

<old>...</old> (Element von „data”; optional)
Umschließt eine Liste mit den ganzen alten (bzw. ursprünglichen) Daten. Unterhalb dieses Elements muss mindestens ein „olddata”-Element vorkommen.

<olddata>...</olddata> (Element von „old”)
Enthält jeweils einen alten Eintrag (bzw. dessen Daten). Verhält sich ansonsten genauso, wie „current”.

<number>...</number> (Element von „current” und „olddata”)
Enthält einen numerischen Wert

<string>...</string> (Element von „current” und „olddata”)
Enthält eine Zeichenkette. Weil das Modul zum Einlesen der Datei Schwierigkeiten hat, bestimmte Sonderzeichen einzulesen, müssen diese kodiert werden. In der deutschen Sprache wäre es z.B. die Umlaute und „ß”.
Bei diesen Zeichen handelt es sich konkret um alle Zeichen, deren ASCII-Wert kleiner als 39 oder größer als 126 sind. Weiterhin müssen die Zeichen „<” und „>” kodiert werden.
Die Kodierung gestaltet sich so, dass deren Unicode-Wert in HTML-Notation transformiert wird.
Beispiel: Aus „Löschen” wird „Löschen”.