Name der Programmiersprache: ASCL very simple calendar language

Domäne: Objekt-Orientiert

Gruppenmitglieder: Steffen Anhäuser, Jan-Erik Menzel, Melina Morch, Lukas Hofmann

==========

Beschreibung der Domäne

Beschreiben Sie die Domäne, für die Sie Ihre Programmiersprache entwickeln möchten.
Die Programmiersprache soll für die Objektorientierte Domäne entwickelt werden. Daher soll eine grundlegende, gut überschaubare Klassenhierachie vorhanden sein mit der sich die Problemstellung behandelt lässt.

Welche Probleme soll sie in der Domäne lösen?
Studiumsplanung im speziellen und eine robustes Framework zur Implementierung von Daten, Uhrzeit, Übungen und Events.
Das Einlesen und daraufhin Parsen von verschiedenen Dateiformaten ohne großen Aufwand.
Die Sprache soll einen dynamischen Austausch von Dateien mit anderen Nutzern ermöglichen und einen Workflow wie in typischen Versionsverwaltungen ermöglichen.

Qualitätsziele

Die Sprache soll sich qualitativ durch lesbar und 'sprechenden' Code auszeichnen, d.h. der User der grundlegende Programmierkenntnisse in Sprachen wie Java/C# etc. besitzt sollte keine Probleme haben den Code beim lesen direkt zu verstehen.
Es werden Sonderzeichen/Symbole zur Strukturierung genutzt und Blöcke zu erkennen, weiterhin ist der Code expressiv, also wenig Zeilen verursachen viel Wirkung.

Robustheit:

Welche Bedeutung hat Robustheit im Kontext Ihrer Sprache?
Wie ist Robustheit hier definiert und inwiefern ist Ihre Sprache robust, das heißt also wie werden unerwartete Aktionen und Zustände behandelt?
Defaultcase ist mächtig, dynamische Typisierung, enges Problemfeld führt zu präzisen Eingaben

Modularität:

Welche Aspekte Ihrer Sprache sind modular?
Konzeptuell besteht die Sprache aus wenigen Blöcken mit spezieller Anwendung. Aus diesen setzt sich alles andere zusammen.

Mindestens ein weiteres eigenes Ziel:

Beschreiben Sie mindestens ein weiteres Ziel, das Ihre Programmiersprache verfolgt.
Beispiele: Effizienz, Verständlichkeit, Ausdrucksstärke, Anpassbarkeit, Korrektheit, Überprüfbarkeit, …

Überprüfbarkeit:
Durch die dynamische Typisierung der Attribute/Parameter kann erst während der "run-time" geprüft werden, ob ein Typfehler entsteht
Jedoch lässt sich die Syntax im (debugging)complie Vorgang prüfen, ob die Syntax den grundlegenden Regeln der Sprache entspricht.

Primitive Type: Int, Double, Boolean, String
Zusammengesetzte Typen: Liste, Files, Date/Tag, Event/Task, Woche, Monat, TimeWindow als allgemeines

Modellierungsmodell, User
Kontrollfluss: git-like

CODE SNIPPET IDEA

Anlegen neue Vorlesung:

Deklaration von Instanzen eines Objektes

```
TimeWindow sose := new TimeWindow(Start.Date=20-04-2020,End.Date=17-04-2020); Event exp := new Event(Title="Vorlesung Forgeschrittene Konzepte der Programmierung", TimeWindow=sose,Weekday=WEDNESDAY,BeginTime=10:15, EndTime=11:45, weekly=true);
```

Bedingungen

```
if (sose.Start.Date = "20-04-2020") { System.out.println("Stimmt"); } elseIf (sose.Start.Date < "20-04-2020") { System.out.println("Sie sind in einem einem FS vor SoSe2020"); } elseIf (sose.Start.Date > "20-04-2020") { } else { System.out.println("Sie sind in einem einem FS nach SoSe2020"); }
```

Schleife

```
TimeWindow sose2[] := new TimeWindow()[5]; for (int i = 0; i < 5; i++) { // Hochzählen der Sommersemester sose2[i].Start.Date = sose.Start.Date + i; sose2[i].End.Date = sose.End.Date + i; }
```