

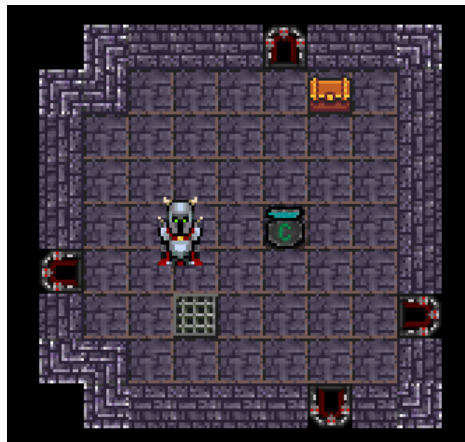
Peer-Feedback, Poster-Sessions und OER in ILIAS-Kursräumen

Finn Amini Kaveh, Carsten Gips (HSBI)

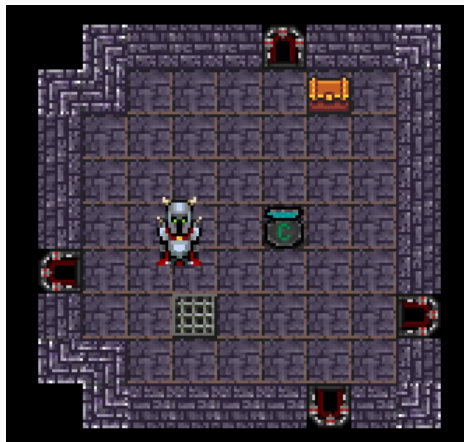
21. November 2023

Unless otherwise noted, this work is licensed under CC BY-SA 4.0.

Lernszenario "Programmiermethoden": Wir entwickeln ein Spiel



Lernszenario "Programmiermethoden": Wir entwickeln ein Spiel



Bearbeitung der
Aufgabe

Abgabe der
Lösung im ILIAS

Vorstellung der
Lösung im
Praktikum

Bewertung/
Feedback durch
Lehrende

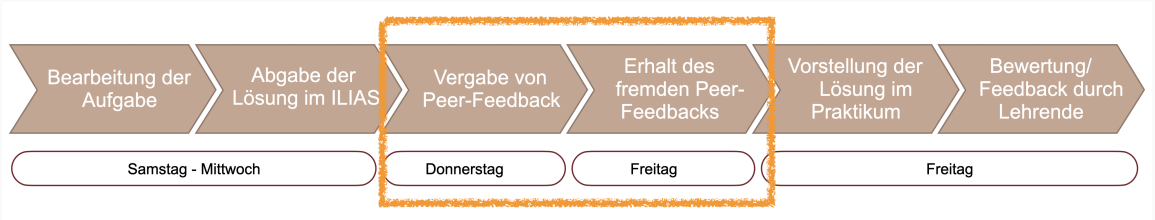
Samstag - Donnerstag

Freitag

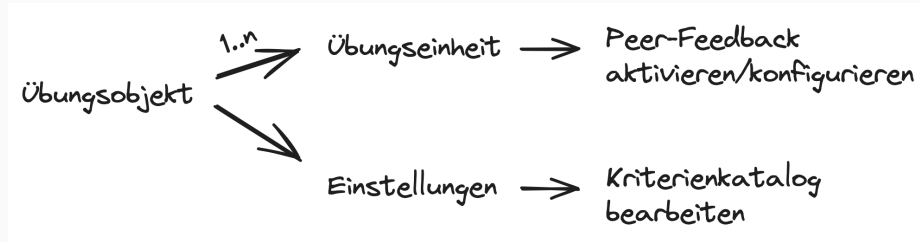
Peer-Feedback zu Übungsaufgaben im ILIAS

Ziele Peer-Feedback: Studierende sollen ...

- Fremde Lösungen (Code) lesen lernen
- Fremde Konzepte bewerten lernen
- Anregungen für ihre eigenen Lösungen bekommen (Spieleentwicklung!)



Peer-Feedback: Kriterienkataloge im Übungsobjekt



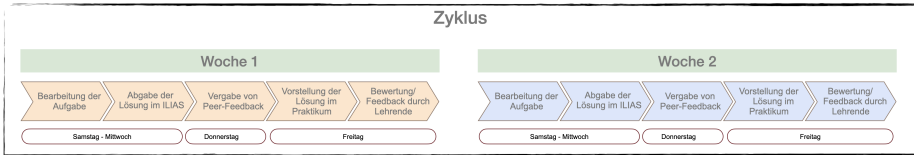
Beispiele für Review-Fragen:

- Wie gut können Sie die Modellierung nachvollziehen? (5-Sterne-Bewertung)
- Was gefällt Ihnen an der Modellierung besonders? (Text)
- Wie könnte die Modellierung verbessert werden? (Text)
- Beurteilen Sie die Dokumentation des Codes und geben Sie Verbesserungshinweise. (Text)
- Kein Review – es handelt sich um die Abgabe meines Teams. (Erfüllt Ja/Nein)

Peer-Feedback: Lessons Learned

- **Fragen:** Lieber Freitext statt Checkboxes bzw. 5-Sterne-Bewertung nutzen
- **ILIAS:**
 - Peer-Feedback lässt sich nur für Einzelabgaben konfigurieren
 - Keine Nachfrist oder individuelle Abgabe möglich
 - Kriterienkataloge lassen sich nicht kopieren
- **Organisation:**
 - Zusätzliche Bearbeitungszeit für Peer-Feedback notwendig
 - Peer-Review = zusätzliche "Abgabe" (aus Studierenden-Perspektive)
 - Höherer Workload für die Studierenden

Mehrwöchige Arbeitszyklen: Orientierung im ILIAS



This screenshot shows the ILIAS interface for a course. The main content area displays a calendar view for 'Praktikum: Aktueller Zyklus: Zyklus 1'. A date 'Aktiv bis: 12. Apr 2023, 00:00' is circled in orange. Below the calendar, there are sections for 'Woche 03: Zyklus 1: Konzept' and 'Woche 04: Zyklus 1: Implementierung', each with a list of activities and their scheduled times.

- Woche 03: Zyklus 1: Konzept**
 - Konzept: 20.04. - 08:00 Uhr
 - Peer-Feedback: 21.04. - 08:00 Uhr
 - Praktikum: 21.04.
- Woche 04: Zyklus 1: Implementierung**
 - Implementierung: 27.04. - 08:00 Uhr
 - Peer-Feedback: 28.04. - 08:00 Uhr
 - Praktikum: 28.04.



This screenshot shows the ILIAS interface for a course, featuring a welcome message and a calendar view. The main content area displays a calendar view for 'Praktikum: Aktueller Zyklus: Zyklus 1'. A notification 'Vorsicht als Mitglied' is visible at the top right. Below the calendar, there are sections for 'Woche 03: Zyklus 1: Konzept' and 'Woche 04: Zyklus 1: Implementierung', each with a list of activities and their scheduled times.

Herzlich willkommen zu Programmiermethoden im S23! "Weniger schlecht programmieren" :-)

- Woche 03: Zyklus 1: Konzept**
 - Konzept: 20.04. - 08:00 Uhr
 - Peer-Feedback: 21.04. - 08:00 Uhr
 - Praktikum: 21.04.
- Woche 04: Zyklus 1: Implementierung**
 - Implementierung: 27.04. - 08:00 Uhr
 - Peer-Feedback: 28.04. - 08:00 Uhr
 - Praktikum: 28.04.

Poster-Galerie im Modul "Künstliche Intelligenz"

Seite einrichten → Spalten-Layout → Poster im "PNG"-Format als "Bild/Audio/Video" einfügen → Medienobjekt: "Vollbild" aktivieren

The screenshot displays the ILIAS LMS interface for the 'Künstliche Intelligenz' module. The top navigation bar shows the course path: 'Wintersemester 2022/23 > IFM-S/W - Liste 1 - Künstliche Intelligenz - Gips - WS2022/23 > Poster Galerie (Projekt 1)'. The main content area is a grid of posters:

- Poster 1: Lösung des Stundenplanproblems mit dem genetischen Algorithmus** (Genetic Algorithm for Timetable Problem)
- Poster 2: Stundenplanproblem** (Timetable Problem) - Includes sections for '1. Vorgehen', '2. Modellierung', '3. Ergebnis', and '4. Fazit'.
- Poster 3: Stundenplanung** (Timetable Planning) - Includes sections for 'Vorgehen' and 'Modellierung'.
- Poster 4: Stundenplangenerierung mittels Genetischer Algorithmen** (Timetable Generation using Genetic Algorithms) - Includes sections for 'Vorgehen', 'Modellierung des Problems', 'Fitness eines Individuums', and 'Ergebnis'.
- Poster 5: EINFÜHRUNG** (Introduction) - Discusses the problem and the genetic algorithm approach.
- Poster 6: ZIEL** (Goal) - States the objective of the algorithm.
- Poster 7: METHODE** (Method) - Describes the genetic algorithm process.
- Poster 8: ANALYSE** (Analysis) - Shows a graph of fitness over generations.
- Poster 9: ERGEBNIS** (Result) - Shows the final timetable solution.
- Poster 10: STUDENTENPLANUNG MIT GENETISCHEN ALGORITHMEN** (Timetable Planning with Genetic Algorithms) - A summary poster.
- Poster 11: Das Stundenplan-Problem als CSP** (The Timetable Problem as a CSP) - Discusses the problem as a Constraint Satisfaction Problem.

Vielen Dank an die Digi-Scouts vom DigikoS-Projekt für die Unterstützung bei der technischen Umsetzung im ILIAS!

Beispiel Lernmodul Compilerbau

Lernmodul Startseite

IFM 5.21: COMPILERBAU (WINTER 2023/24)

```
#include <stdio.h>
int main() {
    printf("hello world!");
}
```

gcc

lexer

parser

semantische analyse

zweischencode

compiler

Kursbeschreibung

Der Compiler ist das wichtigste Werkzeug in der Informatik. In der Königsdisziplin der Informatik schließt sich der Kreis, hier kommen die unterschiedlichen Algorithmen und Datenstrukturen und Programmiersprachenkonzepte zur Anwendung.

In diesem Modul geht es um ein grundlegendes Verständnis für die wichtigsten Konzepte im Compilerbau. Wir schauen uns dazu relevante Tools und Frameworks an und setzen diese bei der Erstellung eines kleinen Compiler-Frontends für `Mini-Python` ein.

Überblick Modulinhalte

- Lexikalische Analyse: Scanner/lexer
 - Reguläre Sprachen
 - Generierung mit ANTLR
- Syntaxanalyse: Parser
 - Kontextfreie Grammatiken (CFG)
 - LL-Parser (Top-Down-Parser)
 - Generierung mit ANTLR
- Semantische Analyse: Attributierte Grammatiken und Symboltabellen
 - Namen und Scopes
 - Typen, Klassen, Polymorphie
- Zwischencode: Intermediate Representation (IR), Builder
- Interpreter: AST-Traversierung

Lernmodul Sitzung

ANTLR GENERIEREN

Der Compiler ist das wichtigste Werkzeug in der Informatik. In der Königsdisziplin der Informatik schließt sich der Kreis, hier kommen die unterschiedlichen Algorithmen und Datenstrukturen und Programmiersprachenkonzepte zur Anwendung.

In diesem Modul geht es um ein grundlegendes Verständnis für die wichtigsten Konzepte im Compilerbau. Wir schauen uns dazu relevante Tools und Frameworks an und setzen diese bei der Erstellung eines kleinen Compiler-Frontends für `Mini-Python` ein.

Überblick Modulinhalte

- Lexikalische Analyse: Scanner/lexer
 - Reguläre Sprachen
 - Generierung mit ANTLR
- Syntaxanalyse: Parser
 - Kontextfreie Grammatiken (CFG)
 - LL-Parser (Top-Down-Parser)
 - Generierung mit ANTLR
- Semantische Analyse: Attributierte Grammatiken und Symboltabellen
 - Namen und Scopes
 - Typen, Klassen, Polymorphie
- Zwischencode: Intermediate Representation (IR), Builder
- Interpreter: AST-Traversierung

Lexer: Erzeugen eines Token-Stroms aus einem Zeichenstrom

Definition wichtiger Begriffe

Typische Motive für Erzeugung von Token

hello world

Hinweis zur Grammatik (Regel)

ANTLR schreiben

"hello world" überlesen und ausführen

Generierte Tabellen und Klassen

Beibehaltung der Ausgaben

ANTLR-Grammatik für die Lexer-Generierung

lexend und non-greedy Lexer-Regeln

Verhalten des Lesers: 1. Längster Match

Verhalten des Lesers: 2. Reihenfolge

Verhalten des Lesers: 3. Non-greedy Regeln

Aktionen und Aktionen

Aktionen der Token-Stream

Aktionen mit dem Lexer-Regel

Wrap-up

werden. Diese Aktionen müssen in der Zielsprache formuliert werden, da sie in die generierte Lexerklasse in die jeweiligen Methoden eingebettet werden.

Videos (Youtube)

- VL Lexer mit ANTLR
- Demo ANTLR Basics
- Demo Verhalten Lexer-Regeln
- Demo Lexer-Regeln mit Aktionen

Videos (HöR-Medienportal)

- VL Lexer mit ANTLR

Lernziele

- (K1) Lexer-Regeln in ANTLR formulieren und einsetzen
- (K2) Verhalten des Lesers: längste Matches, Reihenfolge
- (K3) Nutzung von Lexer-Aktionen

Lexer: Erzeugen eines Token-Stroms aus einem Zeichenstrom

Aus dem Eingabe-`quell-text`



Unless otherwise noted, this work is licensed under CC BY-SA 4.0.

See github.com/cagix/dlk23 for sources, slides and handout.