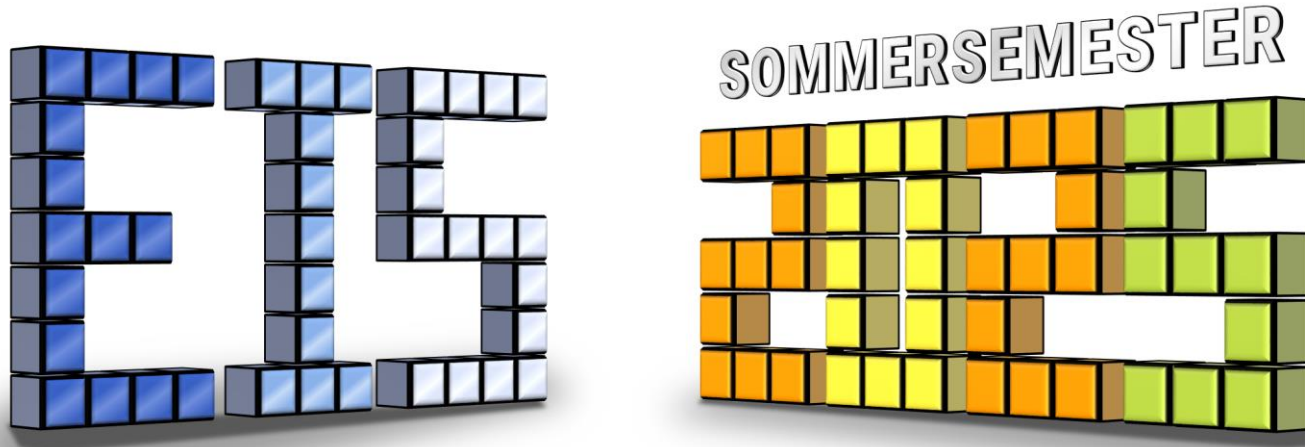


EINFÜHRUNG IN DIE SOFTWAREENTWICKLUNG

Sommersemester 2025



Foliensatz #1

Organisatorisches

Michael Wand
Institut für Informatik
Michael.Wand@uni-mainz.de



Who's Who



Vorlesung

Michael WAND

Raum 03-334

Michael.Wand@uni-mainz.de



Übungen

Christian ALI MEHMETI-GÖPEL

Raum 03-333

chalimeh@uni-mainz.de

Zentraltutorium

Paul Becker

Übungsgruppen

**Raphael Eiden, Simon Laubersheimer, David Ludat,
Tam-Anh Tran, Dennis Scheck, Leon Steiner**

Übersicht

Inhalt heute

- Organisatorisches
 - Konzept der Veranstaltung
 - Vorlesung, Übungen, Prüfung, etc.

Organisatorisches

Web-Systeme

„Jogustine“

<https://jogustine.uni-mainz.de>

- Anmeldung zu Vorlesungen, Prüfungen

„LMS“ / „Moodle“

<https://lms.uni-mainz.de>

- Vorlesungsfolien, Übungszettel
- Abgabe der praktischen Übungen

„Teams“

<https://teams.microsoft.com/>

- Abgabegruppen finden
- Diskussionen & Austausch

code: [55yb23b](#)

WWW-Seite

<http://luna.informatik.uni-mainz.de/eis25>

- Aktuelle Infos, zusätzliche Materialien

EIS Komponentenarchitektur

Was gehört alles dazu?

1. Vorlesung
2. Übungsaufgaben
3. Übungsgruppen
4. Zentrales Tutorium
5. Prüfungsmodalitäten

EIS Komponentenarchitektur

Was gehört alles dazu?

1. **VORLESUNG**

- 2. Übungsaufgaben
- 3. Übungsgruppen
- 4. Zentrales Tutorium
- 5. Prüfungsmodalitäten

Vorlesung

Vorlesung

- Dienstags 12:15 – 13:50 (c.t.)
 - 2 × 45 min. Vorlesung pro Woche
 - 5 min. Halbzeitpause
- Hörsaal 00 521N 1 / “Muschel” Hörsaal N1

Folien

- Werden bereitgestellt auf lms.uni-mainz.de
 - I.d.R. am Abend nach der Vorlesung verfügbar

Vorlesung

Vorlesungsvideos

- Aufzeichnung aus dem Sommersemester 2024
 - Zusätzliches Material zur Nachbereitung
 - Formal: kein Ersatz für Vorlesung
 - Gekürzt (keine Rückfragen, keine Memes)
 - Etwas andere Reihenfolge und Inhalte als dieses Semester
 - Inhalte der Präsenzvorlesung zählen für Prüfung
- Verfügbar via Panopto (Link ist auch im LMS)

<https://video.uni-mainz.de/Panopto/Pages/Sessions/List.aspx?folderID=7f17c25c-68b3-47ce-b857-b15600e16b9b>

EIS in 5 Simple Steps...

Was gehört alles dazu?

1. Vorlesung
2. **ÜBUNGSAUFGABEN**
3. Übungsgruppen
4. Zentrales Tutorium
5. Prüfungsmodalitäten

Übungsaufgaben

Übungsaufgaben

- Es gibt Übungsaufgaben alle 1-2 Wochen
 - Übungsaufgaben = Hausaufgaben
 - Erscheinen (spätestens) Montags morgens auf lms.uni-mainz.de
- Abgabe ist Pflicht
 - Man muss alle Übungsblätter bearbeiten...
 - ...und mindestens 50% der Punkte erreichen...
 - ...um „zur Klausur zugelassen zu werden“
 - Klausur macht sonst keinen Sinn!
- Erstes „Übungsblatt“ steht bereits in LMS

Übungsaufgaben

Was lernt man in den Übungen

- Das allermeiste!
- Selbst die Inhalte aus der Vorlesung anwenden

Themen der Übungen

- EIS ist sehr praxisnah
- Übungen in Methoden
 - 1 Woche Bearbeitungszeit
- Softwareprojekte
 - 2 Wochen Bearbeitungszeit
 - Ähnliche Projektthemen wiederholt (Ansätze vergleichen)

Übungsaufgaben

Auf- und Abschreiben

- Abschreiben von Übungsaufgaben ist verboten
 - Nicht von anderen Studis
 - Nicht aus dem Netz
 - Nicht von AI-Assistent generieren lassen (z.B. GPT)
 - Als Berater nützlich, aber erst selbst lösen
- Abschreiben ist blöd
 - Man lernt nur in den Übungen
 - Klausurzulassung gibt keine Note, nur Klausur gibt Note
 - Wenn man nicht geübt hat, fällt man durch
 - Klausur: Maximal 3 Versuche!
- Skillz matter

Übungszettel: Input & Output

Wo finde ich die Übungsaufgaben?

- Veröffentlicht im LMS (Moodle)
 - Neue Übungsaufgaben Montags
 - Erstes „Übungsblatt“ steht schon auf LMS
 - Abgabe

Wo gebe ich die Übungsaufgaben ab?

- Im LMS (Moodle)
- Menüpunkt für Abgabe von Aufgaben
- Abgabefrist: Sonntags, 23:59h
 - Datum steht immer auf dem Aufgabenzettel

Übungsaufgaben

Formaler Ablauf

- Abgabe von Übungsaufgaben in Gruppen
 - 4 Studierende pro Gruppe (evtl. Erhöhung auf 5, dann Nachricht)
 - Die gleichen Punkte für alle in der Gruppe

Die Sache mit den Abgabegruppen...

- Gruppen*abgabe* ist Notlösung
 - Beschränkte Ressourcen für Korrekturen
 - Jede(r) sollte die Aufgaben selbst lösen (können)
- Gruppen*arbeit* ist wichtig!
 - Man lernt am meisten voneinander
 - Lerngruppen formen! Offen sein! Kontakte suchen!

Übungsaufgaben

Wie finde ich Mitstreiter/innen?

- Teams-Kanal „Lerngruppe gesucht“
 - Seien Sie offen und nett zueinander :-)
- Übungsgruppen nächste Woche
 - Spezieller Termin für informellen Kontakt
 - Einfach für Übungsgruppe anmelden, dann Lerngruppe suchen
- „Ich habe meine Freunde/innen mitgebracht!“
 - Wunderbar. Bitte alle für die gleiche Übung anmelden.

Wie gebe ich ab?

Elektronisch

- Textaufgaben
 - Abgabe als *.PDF, *.JPG oder *.PNG
- Programmieraufgaben
 - Als Source Code (*.PY)
 - Das sind die meisten
- Ggf. mehrere Dateien als Archiv (*.ZIP)

Was passiert nach der Abgabe?

Neues Modell seit SoSem 2023

- Keine gewöhnliche Korrektur
 - Persönliche „Abnahme“ der Übungen in 20-Minuten-Slots
 - Eigenes Gerät (Laptop) mitbringen!
 - Sonst Absprache vorher!
- Gespräch mit Tutoren/innen mit **ganzer** Gruppe
 - Nur anwesende bekommen Punkte
 - Alle Mitglieder müssen alle Aufgaben erklären können
 - Individueller Punktabzug bei offensichtlichen Wissenslücken
 - Empfehlung: Bringen Sie Ihre eigene Lösung zur Abgabe mit!
 - Zusätzlich zur Gruppenlösung
 - Man muss eine Lösung erklären können!

Vermeidung Anwesenheitspflicht

Anwesenheitspflicht nicht erzwungen

- Alternative: Opt-Out
 - Rein schriftliche Abgabe ist auf Wunsch möglich
 - Gleiche Maßstäbe bei der Bewertung
- Wir empfehlen diese Variante nicht
 - Persönliches Gespräch ist didaktisch wertvoll
 - Wenn persönliche Umstände dies Erfordern, sprechen Sie mit uns – wir versuchen zu helfen

Vermeidung Anwesenheitspflicht

Schriftliche Abgabe

- Abgabe **muss** auch in 4er-Gruppen erfolgen
- Die Abgabe muss enthalten:
 - Eine Gruppenlösung (wird bewertet)
 - Individuelle Lösungen jedes Gruppenmitglieds
 - Im Anhang zur Gruppenlösung
 - Sollten diese ähnlich sein, muss erklärt werden warum
 - „Gemeinsam erarbeitet“ ist erlaubt, aber Falschangaben sind nicht zulässig
 - Grund: Gleiche Kontrolle der individuellen Leistung muss möglich sein

Vermeidung Anwesenheitspflicht

Wie geht es?

- Falls es nicht anders geht: Anmeldung für rein schriftliche Abgabe via formloser Email an Dozent (Michael.Wand@uni-mainz.de)
 - Frist wie für reguläre Übungsanmeldung (21.04.25/Ostersonntag)
 - Das gilt dann für das gesamte Semester
 - Die Abgaben vor Ort entfallen komplett, keine Anwesenheit
 - Tutorium ohne Abgabe optional
 - In der Zentralübung kann man sich wann immer gewünscht über die Musterlösung informieren
- Begründung in Email wird begrüßt, aber nicht verlangt
 - Feedback hilfreich zur Verbesserung der Lehre

EIS in 5 Simple Steps...

Was gehört alles dazu?

1. Vorlesung
2. Übungsaufgaben
3. **ÜBUNGSGRUPPEN**
4. Zentrales Tutorium
5. Prüfungsmodalitäten

Übungsgruppen

Abgabe in Übungsgruppen

- Es gibt sieben Übungsgruppen
 - Jede Übungsgruppe entspricht 2h-Slot
 - Am ersten Termin (nächste Woche) wird der gesamte 2h-Slot genutzt
 - Anmeldung zu erste Übung noch ohne Gruppeneinteilung
 - Zweck ist auch, Lerngruppen zu finden
 - Wegen Ostern: Montagsgruppe am 28.04! (statt 21.04.)
- Danach: Abnahmen
 - Jede Gruppe spricht 20min-Slot mit Tutor/in ab
 - Vorstellung der Lösungen
 - Bewertung und Diskussion durch Tutor/in

Zweiwöchige Projekte

Übungsaufgaben: Im Wechsel

- 1 Woche Bearbeitungszeit: eher konzeptionell
- 2 Wochen Bearbeitungszeit: eher projektartig

Übungsgruppen bei 2-Wochen Projekten

- Am Ende Abnahmen, wie beschrieben
- Dazwischen Tutorium
 - Typischerweise mit gesamter Gruppe an einem Termin
 - Individuell definiert / gestaltet durch Tutor/in
 - Tutorium ohne Abnahme: Keine Anwesenheitspflicht

Anmeldung für Übung EIP

Achtung, zwei Schritte!

- 1.** Übungsgruppe (Tutor/in) auswählen
 - Zwölf mögliche Gruppen, Auswahl in LMS
 - **Anmeldefrist LMS:** 20.04.2025, 23:59h (Ostersonntag)
 - Erste Tutorien (ohne 20Min-Slots) 21. – 25., 04.2025
- 2.** Lerngruppe bilden aus 4 Studierenden
 - Man kann Mitstreiter/innen im ersten Tutorium finden
 - Auf LMS als Gruppe anmelden
 - Bei der Anmeldung der Abgabegruppe wird direkt der 20-Min. Slot festgelegt!
 - **Anmeldefrist LMS:** 27.04.2025, 23:59h (Sonntag)

Übungsgruppen – Optionen

Sieben Termine für Übungsgruppen (→ LMS)

1. **MO** 16-18h (Raum 04-516)
2. **DI** 16-18h (Raum 04-512)
3. **MI** 12-14h (Raum 04-224)
4. **MI** 14-16h (Raum 04-224)
5. **DO** 14-16h (Raum 04-512)
6. **DO** 16-18h (Raum 04-512)
7. **FR** 10-12h (Raum 04-426)

tlw. 2-3 Gruppen pro Termin – insgesamt 12 Gruppen

Übungsgruppen

| Gruppen Nr | Tutor | Tag | Uhrzeit | Raum | Bemerkung |
|------------|---------------------|-----------|---------|--------|--|
| 1 | Dennis SCHECK | MI | 14-16h | 04-224 | |
| 2 | Dennis SCHECK | DO | 14-16h | 04-512 | |
| 3 | David LUDAT | MO | 16-18h | 04-516 | Ostermontag: Ersatz DI 16-18 (Raum 04-512) |
| 4 | David LUDAT | MI | 14-16h | 04-224 | |
| 5 | Leon STEINER | DO | 14-16h | 04-512 | |
| 6 | Leon STEINER | FR | 10-12h | 04-426 | |
| 7 | Simon LAUBERSHEIMER | DO | 16-18h | 04-512 | |
| 8 | Simon LAUBERSHEIMER | FR | 10-12h | 04-426 | |
| 9 | Raphael EIDEN | MI | 12-14h | 04-224 | |
| 10 | Raphael EIDEN | MI | 14-16h | 04-224 | |
| 11 | Tam-Anh TRAN | DI | 16-18h | 04-512 | |
| 12 | Tam-Anh TRAN | DO | 14-16h | 04-512 | |

EIS in 5 Simple Steps...

Was gehört alles dazu?

1. Vorlesung
2. Übungsaufgaben
3. Übungsgruppen
4. **ZENTRALES TUTORIUM**
5. Prüfungsmodalitäten

Zentrales Tutorium

Zentrales Tutorium

- Termin: Dienstags, 10-12h, C04 Chemie (voraussichtlich!)
 - Erstmalig nächste Woche (22.04.2024)
- Studentischer Tutor Paul Becker
 - ...stellt Lösungen der Übungsaufgaben vor
 - ...gibt Tipps für das nächste Übungsblatt
 - ...gibt Tipps zu Tools, Installation & Programmieren
 - Dadurch mehr Zeit für individuelle Fragen in Übungsgruppen
- Veranstaltung on-site/hybrid
 - On-Site im Raum C04 Chemiehörsaalgeb. („voraussichtlich“!)
 - Video-Streaming auf Teams (+ Aufgezeichnet für min. 1 Woche)

Zentrales Tutorium

Teilnahme wichtig!

- Hintergrundinfos zu praktischen Themen
- Tipps & Fragestunde
- Teilnahme freiwillig, aber stark empfohlen
 - Asynchrone Video-Option bei Terminkonflikten

Übungsaufgaben

- EIS hat sehr anspruchsvolle Übungen!
- Im Zentraltutorium gibt es essentielle Zusatzinformationen!

Asynchronität

Warum Zentraltutorium?

- Am Anfang der Vorlesung müssen wir Grundlagen besprechen
 - Softwareentwurf abstrakt
 - Neue Programmiersprache(n)
- In den Übungen gehen wir aber direkt in die Praxis
 - Programmieren mit QT oder AWT-Swing
 - Tutorial dazu im Zentraltutorium!
 - Erst später in der Vorlesung mit mehr Details/Konzepten

EIS in 5 Simple Steps...

Was gehört alles dazu?

1. Vorlesung
2. Übungsaufgaben
3. Übungsgruppen
4. Zentrales Tutorium

5. PRÜFUNGSMODALITÄTEN

Am Ende gibt's Noten...

Prüfung im Fach E.i.S.

- Abschlussklausur am Ende
 - Vorlesungsfreie Zeit
 - Do, 07. August 2025 09:00-12:00, Hörsaal RW1
 - Note in der Klausur = Endnote
- Es gibt dieses Semester kein „Mid-Term-Exam“

Klausurzulassung

Wer darf an der Klausur teilnehmen?

- Sie müssen mindestens 50% der Punkte erreichen
 - Sonst keine Klausurteilnahme möglich
 - Damit auch kein Bestehen der Veranstaltung möglich
 - Es geht um Qualifikation, nicht Punktesammeln!
- Nur Studierende, die persönlich abgeben bekommen Punkte
 - Ausnahme bis zu 2x möglich, wenn vorher mit Tutor/innen abgesprochen
 - Nur „wichtige“ Gründe (Krankheit, Hochzeit, Pflege etc.)
 - Danach Attest (bzw. entspr. Nachweis) erforderlich

Klausurinhalte

Übungen in EIS zentral

- Wir wollen lernen, Software zu entwickeln
- Projektorientierte Übungen helfen dabei
- Klausur orientiert sich in Teilen sehr eng an den Übungen!

Bottom-Line

- Wenn man alle Übungen selbstständig löst, sollte man leicht eine gute Note in der Klausur bekommen

Wiederholder?

„Aber ich habe die VL bei einem anderen Dozenten gehört und wiederhole nur“

- Wir empfehlen natürlich stark, nochmal die Übungen voll mitzumachen

Der Fairness halber

- Es wird 1-2 Aufgaben geben, bei der man einen Schwerpunkt aus dem vorherigen Semester statt einer übungsnahen Aufgabe wählen kann
 - Wahl steht allen Klausurteilnehmer/innen frei
 - Alternative vergleichbar schwierig

Digitale Methodik in den Geistes- und Kulturwissenschaften

Vorlesung “pur”

- Übungen nicht verpflichtend
- Keine gesonderte Prüfung
 - Statt dessen: Veranstaltung HS Mainz

Empfehlung

- Vorlesung ohne Übungen recht sinnfrei
- Übungsblätter bearbeiten & Übungen besuchen
- Skillz matter. CPs sind Schall & Rauch.

Veranstaltungskonzept

Softwareentwicklung

Einführung in die Softwareentwicklung

- Entwicklung größerer Systeme

Drei Aspekte

- Programmieren & Algorithmen
- Design & Softwarearchitektur
 - Ideen / Konzepte / Muster
 - Programmiersprachen & Techniken
- Projektmanagement und Teamwork

Softwareentwicklung

Einführung in die Softwareentwicklung

- Entwicklung größerer Systeme

Drei Aspekte

- Programmieren & Algorithmen (,,EIP“)
 - Design & Softwarearchitektur (klassisches Informatik Studium)
 - Muster / Ideen / Konzepte
 - Programmiersprachen & Techniken
 - Projektmanagement und Teamwork (,,SE“)
- (,,EIS“)

Einschub: Softwarearchitektur

Persönliche Meinung

Meine Meinung

- SE stark Erfahrungs- und Geschmackssache
 - Programmieren ist Kunst
 - Programmieren ist aber auch eine professionelle Aktivität
 - Breites Wissen hilft beiden Aspekten
- Soziale Kompetenzen von zentraler Bedeutung
 - Die Mehrheit von Softwareprojekten scheitert (statistisch)
 - Gründe: Unklare Ziele, schlechte Kommunikation, mangelnde Erfahrung
 - Sehr komplexes logisches Konstrukt muss von Menschen verstanden werden
 - Empathie viel wichtiger als Tech, Erfahrung unersetzlich

Persönliche Meinung

Meine Meinung

- Wechselwirkung von Technik und Soziologie
 - Gute/schlechte Architektur vermeidet/erzeugt Probleme
 - z.B. Modularisierung
 - Gibt kreative Freiräume
 - Erlaubt es Verantwortungsbereiche zu definieren
 - Ähnliches für Testbarkeit, Wiederverwendung, etc.
 - „Keine technische Lösungen für soziale Probleme“
 - Bessere Programmiersprache hilft nicht, wenn Entwickler/innen nicht miteinander reden
 - „Management“ / strukturiertes Vorgehen immer nötig
- Softwarearchitektur wesentlich für Projekterfolg

Persönliche Meinung

Meine Meinung

- Warnung vor Cargo-Cult
 - „There is no silver bullet“ (Fred Brooks, 1986)
 - Offen für verschiedene Ansätze: Be open minded
 - Je besser man sich auskennt, um so leichter fällt das
- Meine Erfahrung
 - Eigener „Geschmack“ ist gut, Ideologie ist schlecht, Tribalismus ist gefährlich
 - Spannungsfeld: Teamarbeit und eigene Vision – gutes Management & Architektur nötig
 - Gut entworfene Software gibt die richtigen Freiräume

...zurück zum
Konzept

Modulhandbuch

Programmiertechniken

- Statische Typisierung
- Hardwarenahe Programmierung und Techniken für effiziente Abstraktionen
- Funktionsvariablen und Funktionen höherer Ordnung
- Bibliotheksfunkt. der Systemumgebung: Eingabe- / Ausgabe, Netzwerkzugriff
- Graphische Benutzerschnittstellen

Softwareentwurf

- Modularisierung
- Objekte, Klassen und Schnittstellen
- Vererbung, abstrakte Klassen und dynamischer Dispatch
- Abstraktion und Geheimnisprinzip
- Generische Datentypen
- Komponenten- und Klassendiagramme
- Einfache/grundlegende Entwurfs- und Architekturmuster

Die Programmiersprache(n)

Absprache am Institut

- EIS: eine weitere, statisch typisierte Sprache lernen
- Scala empfohlen

“Continuity at Boundaries”

- EIS i.d.R. mit Scala Schwerpunkt
- „Programmiersprachen“ nutzt Scala
- SE oft in JAVA
- DB/CG/HPC/Modellierung mit C++
- Python Everywhere

Mein Konzept

Schwerpunkt

- Lernen, wie man komplexe Systeme baut
 - Verstehen, wo es drauf ankommt
 - Nützliche Ideen / Abstraktionen kennenlernen

Worum es nicht geht

- Kein Kurs über eine Programmiersprache
 - Scala: Veranstaltungsinhalt, aber zweite Priorität
 - C++/Java: Interessant, aber nebensächlich
- Achtung:
 - Typischerweise nimmt Scala etwas mehr Raum ein (Wiederholer!)

Sie müssen lernen

- Die Sprachen Python und Scala
 - Scala in den Übungen etwas weniger breit
- Programmier- und Entwurfstechniken
 - Programmiersprache: Entweder Python oder Scala
 - Mehr Raum in den Übungen für praxisnahe Projekte

Mein Konzept

Wie machen wir es?

- „Ausnahmsweise“ mal praxisnah ;-)
- Wir schauen uns eine größere Aufgabe an
- Wir versuchen Sie mit verschiedenen Konzepten umzusetzen
 - Prozedurale Programmierung
 - Objektorientierte Programmierung
 - Funktionale Programmierung
- Lernen dabei diese Methoden & weitere Ideen kennen
 - Schwerpunkt 1: Programmiertechniken
 - Schwerpunkt 2: Entwurfsmuster (Architecture/Design Patterns)

Neue Programmiersprachen

Beispielcode Vorlesung

- Python + MyPy (statisch typisiert)
- Blick auf Konzepte in Scala
- Gelegentlich: C++ (low-level code)



(Monty Python auf der Suche nach dem Heiligen Gral)

Übungen

- Allgemein freie Wahl
 - Python (mit Type-Annotations) + PySide (Qt für GUIs)
 - Scala (mit AWT/Swing, alternativ)
- Vereinzelte Aufgaben legen Sprache verbindlich fest
 - Es gibt auch vereinzelte Aufgaben in C++, JAVA

Neue Programmiersprachen

Klausur

- Programmieraufgaben – freie Wahl aus
 - Python (mit Type-Annotations)
 - Scala
- Aufgaben zu Konzepten, Wissen, Methoden
 - Es kann auch gezielt zu/nach Codefragmenten in
 - Python *wie auch*
 - Scalagefragt werden.
 - Keine Programmieraufgaben in JAVA/C++!
 - Konzeptionelles Wissen zu allen vier Sprachen klausurrelevant
 - (z.B. „Warum sind C++ Pointer gefährlicher als Python Referenzen?“)

Homogenes Konzept

- Wir lernen Python & Scala in EIP & EIS

Etwas weniger Aufwand bei Sprachen

- Wir machen Scala nicht so intensive

Fokus: Praktische Übungen

- Verschiedene Softwaresysteme in den Übungen
 - Betonung von Kreativität und praktischer Erfahrung
- Übungen sind nicht schwer, aber umfangreich
 - „Die halbe Klausur ist dann schon geschafft.“

Was man lernt...

Architekturmuster: \approx vier größere Ideen

- OOP:
 - Elementares „draw_me()“ Pattern
 - Objektgraphen mit Subtyphierarchien
 - Introspection/meta programming (z.B. Serialisierung)
- Functional: „data flow graphs“,
 - „transform_data()“ statt „draw_me()“
 - Diverse tricks (z.B. on-demand computation)
- Ereignisorientierte Programmierung (GUIs, Server)
- Nebenläufige Systeme: Client-Server Architekturen

Dazu: Umsetzung in Programmiersprachen

Themenliste

(vorläufig/geplant)

Vorlesung 1/5

Inhalt (geplant)

- Organisatorisches
 - Konzept der Veranstaltung
 - Vorlesung, Übungen, Prüfung, etc.
- Programmiersprachen
 - Python + MyPy
 - C/C++
 - Java/Scala
- Der Softwareentwicklungsprozess
 - Probleme & grobe Ansätze
- Versionsverwaltung

Vorlesung 2/5

Inhalt (geplant)

- Beispielprojekt(e)
 - Graphikeditor / GUI
 - (Web-) Server
- Prozedurale Programmierung
 - Analyse + Design
 - Entwurf von abstrakte Datentypen
 - Modularisierung
 - Probleme rein prozeduraler Ansätze
- Objektorientierte Programmierung
 - Konzepte und Methoden
 - „Standard“ Architektur(muster)

Vorlesung 3/5

Inhalt (geplant)

- GUIs und ereignisorientierte Programmierung
 - OO-Komponenten + Events
 - Richtlinien für gute (G)UIs
 - In den Übungen schon früher! (Zentral-)Tutorium besuchen!
- Funktionale Programmierung
 - Code als Daten (auch bei OO)
 - Probleme unseres OO-Designs
 - Das Expression Problem
(am Beispiel unseres Editors)
 - Standard-Architektur(muster)
 - Umfassenden Lösungen für das EP?



Vorlesung 4/5

Inhalt (geplant)

- Hardwarenahe Programmierung
 - Maschinennahe Primitive
 - Zeiger und manuelle Speicherverwaltung
 - Methoden für effiziente Abstraktionen
- Nebenläufige Programmierung
 - Parallele Probleme
 - Services als Abstraktion
 - Beispiel: Webserver
 - Ereignisorientierte Architekturen als Turbo

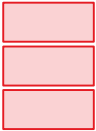






Vorlesung 5/5

Inhalt (geplant)

- Fortgeschrittene Muster
 - Model-View-Controller Architekturen
 - Nicht nur Views: Compiler-Muster
 - Eventspaghetti? Reactive GUIs
 - Heuristiken für gute Architekturen
 - z.B. Form von Interfaces

Plan Übungsaufgaben

Plan für die Übungen (vorläufig)

- | | | | |
|----|--------------------------------------|------------|---|
| 1. | Python, Scala, Java, C++ | (3 Wochen) |  |
| 2. | Einfache GUI Programmierung | (1 Woche) |  |
| 3. | Vektorgraphik prozedural | (2 Wochen) |  |
| 4. | Vektorgraphik objektorientiert + GUI | (2 Wochen) |  |
| 5. | Vektorgraphik funktional | (2 Wochen) |  |
| 6. | Hardwarenahe Programmierung | (1 Woche) |  |
| 7. | Einfacher Webserver | (2 Wochen) |  |

Ablauf der Übungen

Ablauf Übungen

- Ein oder zwei-Wochen-Slots
- Am Ende
 - Abgabe der Übungen (▶)
 - Benotung/Bewertung
 - Teilnahme erforderlich
(außer bei schriftlicher Abgabe)
- In der Mitte (zweiwöchentlich)
 - Übungen zur Beratung (▶)
 - Teilnahme optional, typ. große Gruppe
- Erste Übung: Kennenlernstunde (▶)
 - Große Gruppe

(3 Wochen)

(1 Woche)

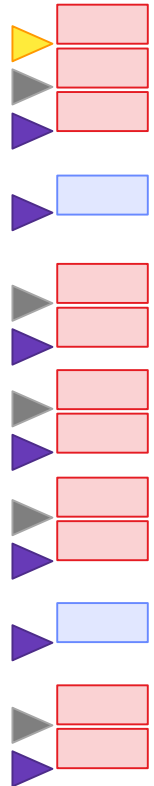
(2 Wochen)

(2 Wochen)

(2 Wochen)

(1 Woche)

(2 Wochen)



Übungsplan (vorläufig/ohne Gewähr)

| Nr. | Datum | Thema | Abgabe von |
|-----|-------------------|---|-------------------|
| 1 | Di, 15. Apr. 2025 | (1) Installation Python, Scala, Java, C++ Übungen mit git „Hello World“ / Prime-Number-Sieve in vier Sprachen | |
| 2 | Di, 22. Apr. 2025 | | |
| 3 | Di, 29. Apr. 2025 | | |
| 4 | Di, 6. Mai 2025 | (2) Einfache GUI Programmierung in PyQt | (1) Install |
| 5 | Di, 13. Mai 2025 | (3) Vektorgraphik-Viewer Prozedural | (2) GUIs |
| 6 | Di, 20. Mai 2025 | <i>(2 wöchig)</i> | |
| 8 | Di, 27. Mai 2025 | (4) Vektorgraphik-Viewer OOP, mit Interaktion (GUI) | (3) Prozedural |
| 7 | Di, 3. Jun. 2025 | <i>(2 wöchig)</i> | |
| 9 | Di, 10. Jun. 2025 | (5) Datenflußarchitektur für Vektorgraphik-Editor (Funktional) | (4) OOP |
| 10 | Di, 17. Jun. 2025 | <i>(2 wöchig)</i> | |
| 11 | Di, 24. Jun. 2025 | (6) Hardwarenahe Programmierung: Image Processing in C++ vs. Python (vs. Scala) | (5) Funktional |
| 12 | Di, 1. Jul. 2025 | (7) Einfacher HTTP-Server in Python | (6) HW-nah |
| 13 | Di, 8. Jul. 2025 | <i>(2 wöchig)</i> | |
| 14 | Di, 15. Jul. 2025 | | (7) Client-Server |



Fragen?

