Bachelorarbeiten

am Institut für Systemsoftware

16. Jänner 2023

Aktuelle Mitarbeiter des SSW

Leitung

Hanspeter Mössenböck

Sekretariat

Karin Gusenbauer

Laura Hoffmann

Systemadministrator

Michael Kaffenda

Stammpersonal / Assistenten

Herbert Prähofer

Markus Weninger

Christoph Pichler

Raphael Mosaner

Sebastian Kloibhofer

Lukas Makor

Aktuelle Mitarbeiter des SSW

Leitung

Hanspeter Mössenböck

Sekretariat

Karin Gusenbauer

Laura Hoffmann

Systemadministrator

Michael Kaffenda

Stammpersonal / Assistenten

Herbert Prähofer

Markus Weninger

Christoph Pichler

Raphael Mosaner

Sebastian Kloibhofer

Lukas Makor

Compilerforschung mit Oracle

Aktuelle Mitarbeiter des SSW

Leitung

Hanspeter Mössenböck

Sekretariat

Karin Gusenbauer

Laura Hoffmann

Systemadministrator

Michael Kaffenda

Stammpersonal / Assistenten

Herbert Prähofer

Markus Weninger

Christoph Pichler

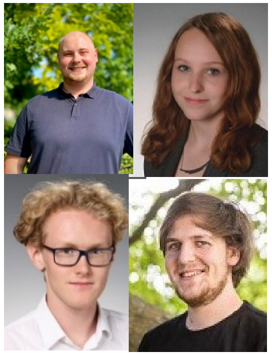
Raphael Mosaner

Sebastian Kloibhofer

Lukas Makor

Compilerforschung mit Oracle

eInformatics



Aktuelle Mitarbeiter des SSW

Leitung

Hanspeter Mössenböck

Sekretariat

Karin Gusenbauer

Laura Hoffmann

Systemadministrator

Michael Kaffenda

Stammpersonal / Assistenten

Herbert Prähofer

Markus Weninger

Christoph Pichler

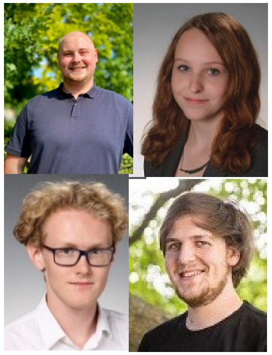
Raphael Mosaner

Sebastian Kloibhofer

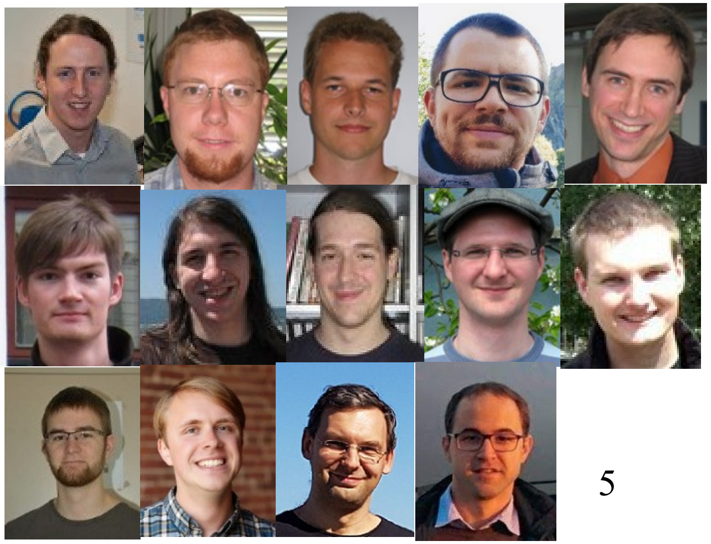
Lukas Makor

Compilerforschung mit Oracle

eInformatics



Oracle Labs



Forschungsbereiche

Compiler und virtuelle Machinen

Graal: hochoptimierender dynamischer Java-Compiler
Truffle: selbstoptimierender AST-Interpreter mit JIT-Compilation

Forschungskooperation mit

Oracle Labs

gute studentische Arbeiten können in Produkte einfließen

Garbage Collection und Memory Monitoring

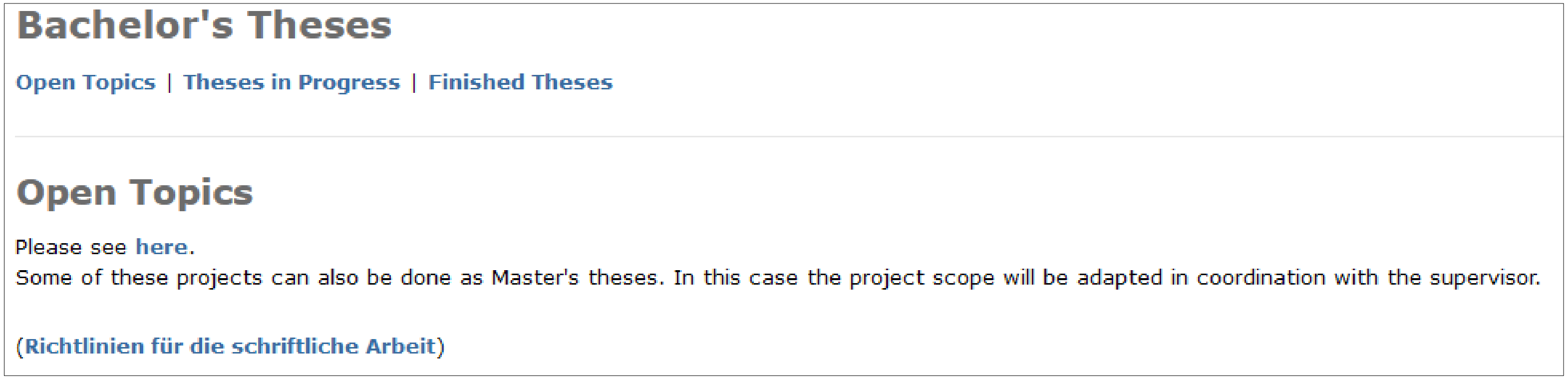
Software Tools
→ eInformatics / JavaWiz: „Graphical Debugger“
→ Xaminer: Online Klausursystem

Laufende/abgeschlossene Bachelorarbeiten

https://ssw.jku.at/Teaching/Projects/

Laufende/abgeschlossene Bachelorarbeiten

https://ssw.jku.at/Teaching/Projects/



Bachelor's Theses

Open Topics | Theses in Progress | Finished Theses

Open Topics

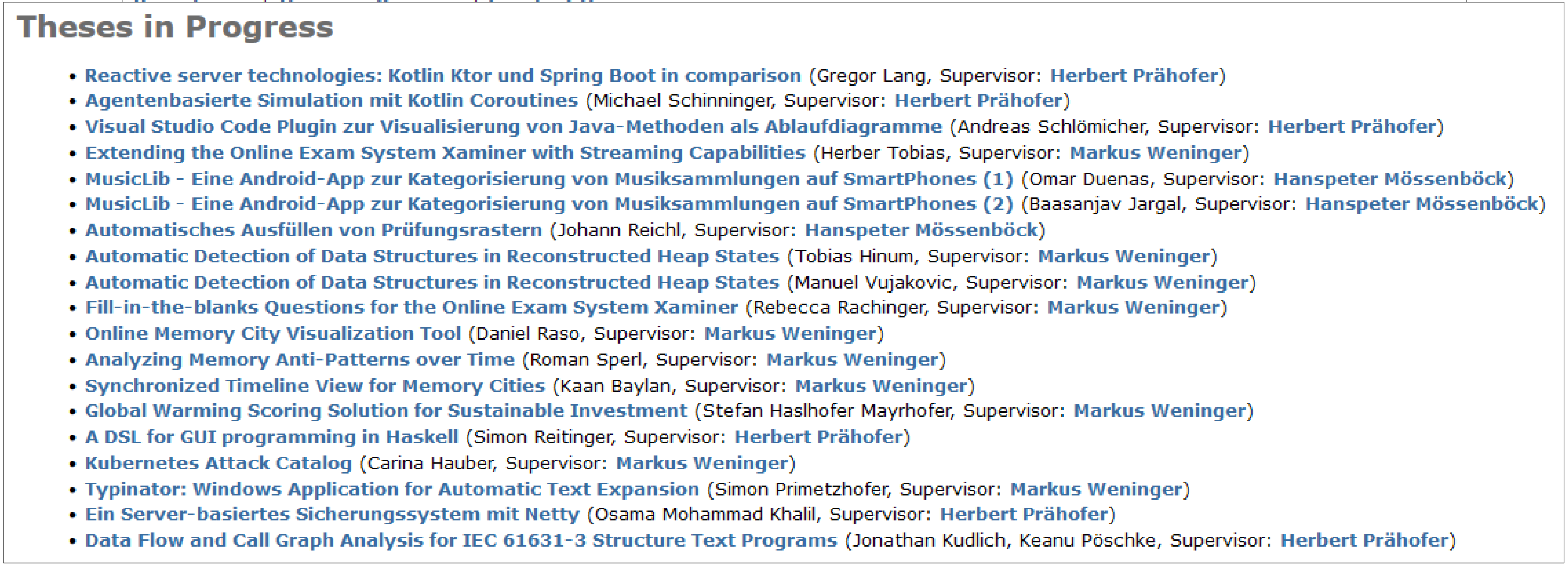
Please see here.
Some of these projects can also be done as Master's theses. In this case the project scope will be adapted in coordination with the supervisor.

(Richtlinien für die schriftliche Arbeit)

Laufende/abgeschlossene Bachelorarbeiten

https://ssw.jku.at/Teaching/Projects/

Bachelor's Theses

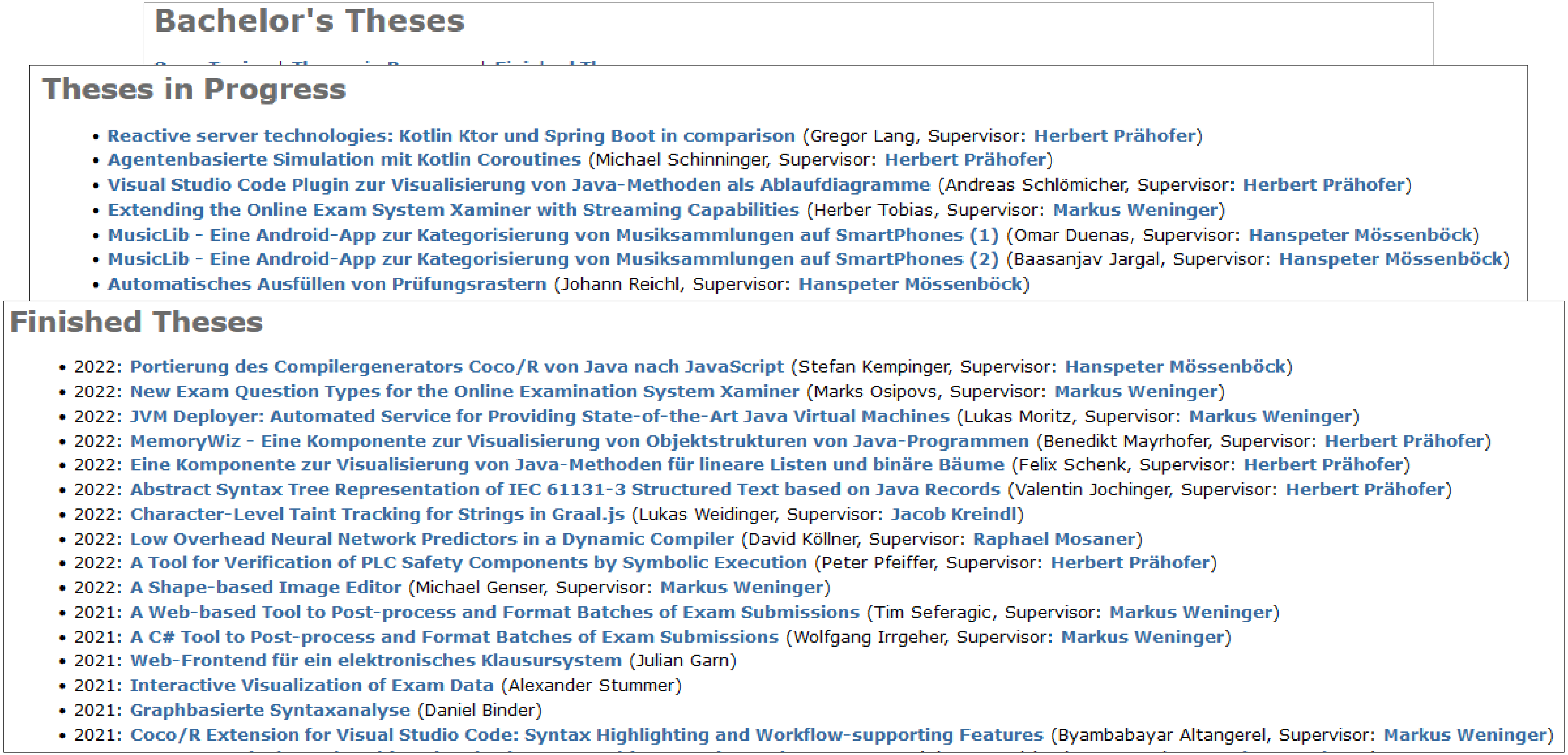


Theses in Progress

- **Reactive server technologies: Kotlin Ktor und Spring Boot in comparison** (Gregor Lang, Supervisor: **Herbert Prähofer**)
- **Agentenbasierte Simulation mit Kotlin Coroutines** (Michael Schinninger, Supervisor: **Herbert Prähofer**)
- **Visual Studio Code Plugin zur Visualisierung von Java-Methoden als Ablaufdiagramme** (Andreas Schlömicher, Supervisor: **Herbert Prähofer**)
- **Extending the Online Exam System Xaminer with Streaming Capabilities** (Herber Tobias, Supervisor: **Markus Weninger**)
- **MusicLib - Eine Android-App zur Kategorisierung von Musiksammlungen auf SmartPhones (1)** (Omar Duenas, Supervisor: **Hanspeter Mössenböck**)
- **MusicLib - Eine Android-App zur Kategorisierung von Musiksammlungen auf SmartPhones (2)** (Baasanjav Jargal, Supervisor: **Hanspeter Mössenböck**)
- **Automatisches Ausfüllen von Prüfungsrastern** (Johann Reichl, Supervisor: **Hanspeter Mössenböck**)
- **Automatic Detection of Data Structures in Reconstructed Heap States** (Tobias Hinum, Supervisor: **Markus Weninger**)
- **Automatic Detection of Data Structures in Reconstructed Heap States** (Manuel Vujakovic, Supervisor: **Markus Weninger**)
- **Fill-in-the-blanks Questions for the Online Exam System Xaminer** (Rebecca Rachinger, Supervisor: **Markus Weninger**)
- **Online Memory City Visualization Tool** (Daniel Raso, Supervisor: **Markus Weninger**)
- **Analyzing Memory Anti-Patterns over Time** (Roman Sperl, Supervisor: **Markus Weninger**)
- **Synchronized Timeline View for Memory Cities** (Kaan Baylan, Supervisor: **Markus Weninger**)
- **Global Warming Scoring Solution for Sustainable Investment** (Stefan Haslhofer Mayrhofer, Supervisor: **Markus Weninger**)
- **A DSL for GUI programming in Haskell** (Simon Reitinger, Supervisor: **Herbert Prähofer**)
- **Kubernetes Attack Catalog** (Carina Hauber, Supervisor: **Markus Weninger**)
- **Typinator: Windows Application for Automatic Text Expansion** (Simon Primetzhofer, Supervisor: **Markus Weninger**)
- **Ein Server-basiertes Sicherungssystem mit Netty** (Osama Mohammad Khalil, Supervisor: **Herbert Prähofer**)
- **Data Flow and Call Graph Analysis for IEC 61631-3 Structure Text Programs** (Jonathan Kudlich, Keanu Pöschke, Supervisor: **Herbert Prähofer**)

Laufende/abgeschlossene Bachelorarbeiten

https://ssw.jku.at/Teaching/Projects/



Bachelor's Theses

Theses in Progress

- Reactive server technologies: Kotlin Ktor und Spring Boot in comparison (Gregor Lang, Supervisor: Herbert Prähofer)
- Agentenbasierte Simulation mit Kotlin Coroutines (Michael Schinninger, Supervisor: Herbert Prähofer)
- Visual Studio Code Plugin zur Visualisierung von Java-Methoden als Ablaufdiagramme (Andreas Schlömicher, Supervisor: Herbert Prähofer)
- Extending the Online Exam System Xaminer with Streaming Capabilities (Herber Tobias, Supervisor: Markus Weninger)
- MusicLib - Eine Android-App zur Kategorisierung von Musiksammlungen auf SmartPhones (1) (Omar Duenas, Supervisor: Hanspeter Mössenböck)
- MusicLib - Eine Android-App zur Kategorisierung von Musiksammlungen auf SmartPhones (2) (Baasanjav Jargal, Supervisor: Hanspeter Mössenböck)
- Automatisches Ausfüllen von Prüfungsrastern (Johann Reichl, Supervisor: Hanspeter Mössenböck)

Finished Theses

- 2022: Portierung des Compilergenerators Coco/R von Java nach JavaScript (Stefan Kempinger, Supervisor: Hanspeter Mössenböck)
- 2022: New Exam Question Types for the Online Examination System Xaminer (Marks Osipovs, Supervisor: Markus Weninger)
- 2022: JVM Deployer: Automated Service for Providing State-of-the-Art Java Virtual Machines (Lukas Moritz, Supervisor: Markus Weninger)
- 2022: MemoryWiz - Eine Komponente zur Visualisierung von Objektstrukturen von Java-Programmen (Benedikt Mayrhofer, Supervisor: Herbert Prähofer)
- 2022: Eine Komponente zur Visualisierung von Java-Methoden für lineare Listen und binäre Bäume (Felix Schenk, Supervisor: Herbert Prähofer)
- 2022: Abstract Syntax Tree Representation of IEC 61131-3 Structured Text based on Java Records (Valentin Jochinger, Supervisor: Herbert Prähofer)
- 2022: Character-Level Taint Tracking for Strings in Graal.js (Lukas Weidinger, Supervisor: Jacob Kreindl)
- 2022: Low Overhead Neural Network Predictors in a Dynamic Compiler (David Köllner, Supervisor: Raphael Mosaner)
- 2022: A Tool for Verification of PLC Safety Components by Symbolic Execution (Peter Pfeiffer, Supervisor: Herbert Prähofer)
- 2022: A Shape-based Image Editor (Michael Genser, Supervisor: Markus Weninger)
- 2021: A Web-based Tool to Post-process and Format Batches of Exam Submissions (Tim Seferagic, Supervisor: Markus Weninger)
- 2021: A C# Tool to Post-process and Format Batches of Exam Submissions (Wolfgang Irrgeher, Supervisor: Markus Weninger)
- 2021: Web-Frontend für ein elektronisches Klausursystem (Julian Garn)
- 2021: Interactive Visualization of Exam Data (Alexander Stummer)
- 2021: Graphbasierte Syntaxanalyse (Daniel Binder)
- 2021: Coco/R Extension for Visual Studio Code: Syntax Highlighting and Workflow-supporting Features (Byambabayar Altangerel, Supervisor: Markus Weninger)

Offene Themen

https://ssw.jku.at/Teaching/Projects/open.html

Offene Themen

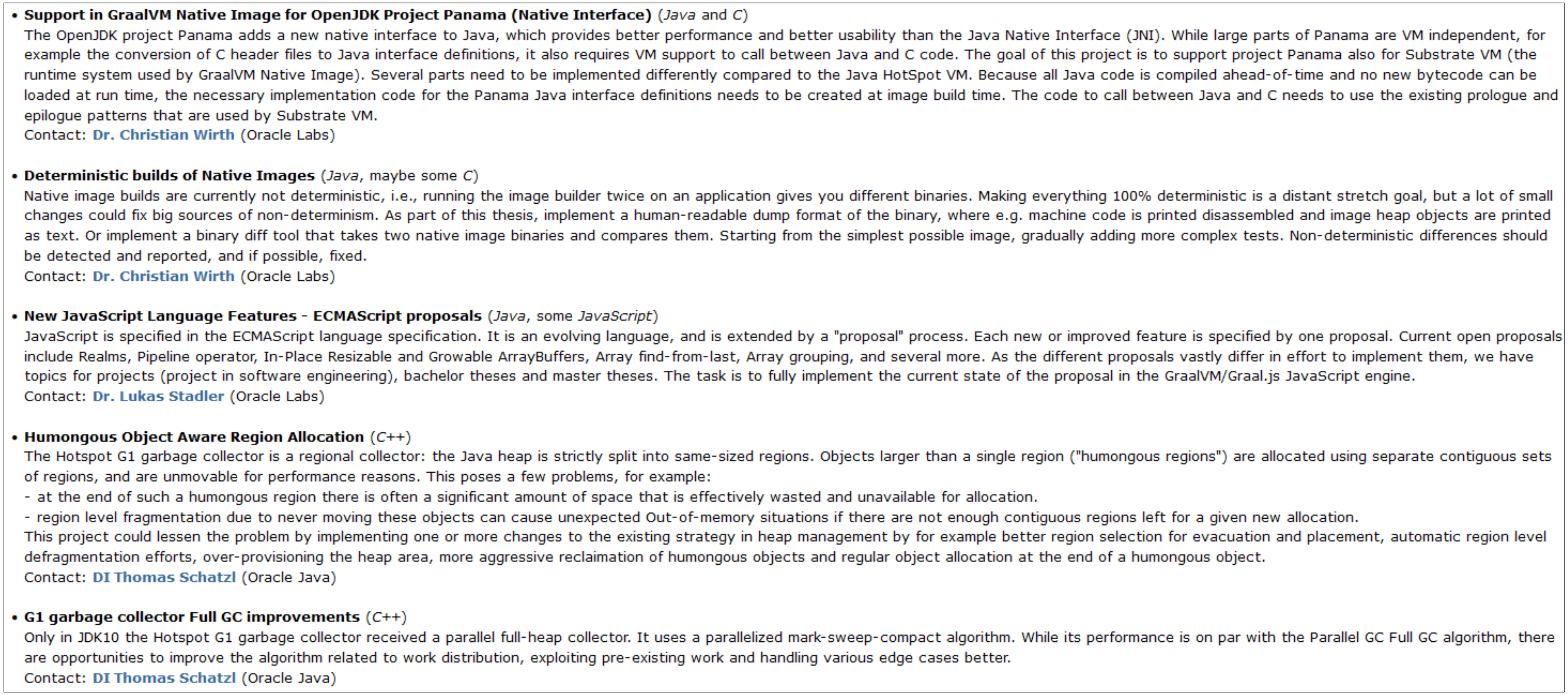
https://ssw.jku.at/Teaching/Projects/open.html

- Es kommen laufend neue Themen hinzu, Liste nicht immer zu 100% aktuell
- Einfach direkt bei Assistenten / Profs nachfragen
- Auch eigene Themen möglich

Offene Themen

https://ssw.jku.at/Teaching/Projects/open.html

- Es kommen laufend neue Themen hinzu, Liste nicht immer zu 100% aktuell
- Einfach direkt bei Assistenten / Profs nachfragen
- Auch eigene Themen möglich



- **Support in GraalVM Native Image for OpenJDK Project Panama (Native Interface)** (*Java* and *C*)
  The OpenJDK project Panama adds a new native interface to Java, which provides better performance and better usability than the Java Native Interface (JNI). While large parts of Panama are VM independent, for example the conversion of C header files to Java interface definitions, it also requires VM support to call between Java and C code. The goal of this project is to support project Panama also for Substrate VM (the runtime system used by GraalVM Native Image). Several parts need to be implemented differently compared to the Java HotSpot VM. Because all Java code is compiled ahead-of-time and no new bytecode can be loaded at run time, the necessary implementation code for the Panama Java interface definitions needs to be created at image build time. The code to call between Java and C needs to use the existing prologue and epilogue patterns that are used by Substrate VM.
  Contact: **Dr. Christian Wirth** (Oracle Labs)

- **Deterministic builds of Native Images** (*Java*, maybe some *C*)
  Native image builds are currently not deterministic, i.e., running the image builder twice on an application gives you different binaries. Making everything 100% deterministic is a distant stretch goal, but a lot of small changes could fix big sources of non-determinism. As part of this thesis, implement a human-readable dump format of the binary, where e.g. machine code is printed disassembled and image heap objects are printed as text. Or implement a binary diff tool that takes two native image binaries and compares them. Starting from the simplest possible image, gradually adding more complex tests. Non-deterministic differences should be detected and reported, and if possible, fixed.
  Contact: **Dr. Christian Wirth** (Oracle Labs)

- **New JavaScript Language Features - ECMAScript proposals** (*Java*, some *JavaScript*)
  JavaScript is specified in the ECMAScript language specification. It is an evolving language, and is extended by a "proposal" process. Each new or improved feature is specified by one proposal. Current open proposals include Realms, Pipeline operator, In-Place Resizable and Growable ArrayBuffers, Array find-from-last, Array grouping, and several more. As the different proposals vastly differ in effort to implement them, we have topics for projects (project in software engineering), bachelor theses and master theses. The task is to fully implement the current state of the proposal in the GraalVM/Graal.js JavaScript engine.
  Contact: **Dr. Lukas Stadler** (Oracle Labs)

- **Humongous Object Aware Region Allocation** (*C++*)
  The Hotspot G1 garbage collector is a regional collector: the Java heap is strictly split into same-sized regions. Objects larger than a single region ("humongous regions") are allocated using separate contiguous sets of regions, and are unmovable for performance reasons. This poses a few problems, for example:
  - at the end of such a humongous region there is often a significant amount of space that is effectively wasted and unavailable for allocation.
  - region level fragmentation due to never moving these objects can cause unexpected Out-of-memory situations if there are not enough contiguous regions left for a given new allocation.
  This project could lessen the problem by implementing one or more changes to the existing strategy in heap management by for example better region selection for evacuation and placement, automatic region level defragmentation efforts, over-provisioning the heap area, more aggressive reclaimation of humongous objects and regular object allocation at the end of a humongous object.
  Contact: **DI Thomas Schatzl** (Oracle Java)

- **G1 garbage collector Full GC improvements** (*C++*)
  Only in JDK10 the Hotspot G1 garbage collector received a parallel full-heap collector. It uses a parallelized mark-sweep-compact algorithm. While its performance is on par with the Parallel GC Full GC algorithm, there are opportunities to improve the algorithm related to work distribution, exploiting pre-existing work and handling various edge cases better.
  Contact: **DI Thomas Schatzl** (Oracle Java)