JOHANNES KEPLER
UNIVERSITY LINZ

Bachelor's Thesis

Online Memory City Visualization Tool

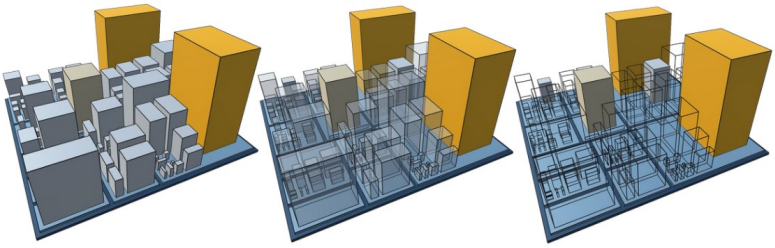
Dipl.-Ing. Dr. Markus Weninger, BSc
Institute for System Software
P +43-732-2468-4361
markus.weninger@jku.at

Student: Daniel Rašo

Advisor: Dipl.-Ing. Dr. Markus Weninger, BSc

Start date: February 2022

Memory Cities, are a technique to visualize an application's heap memory evolution over time using the Software City metaphor. While this metaphor is typically used to visualize static artifacts of a software system such as class hierarchies, we use it to visualize the dynamic memory behavior of an application. In our approach, heap objects can be grouped by multiple properties such as their types or their allocation sites. The resulting object groups are visualized as buildings arranged in districts, where the size of a building corresponds to the number of heap objects or bytes it represents. Continuously updating the city over time creates the immersive feeling of an evolving city. This can be used to detect and analyze memory leaks, i.e., to search for suspicious growth behavior. Memory cities further utilize various visual attributes to ease this task. For example, they highlight strongly growing buildings using color, while making less suspicious buildings semi-transparent.



Currently, Memory Cities is a standalone application developed in Unity, with a JSON-based input file format for easy data import from external tools. Typically, we use Memory Cites in conjunction with AntTracks, a trace-based memory monitoring tool that is able to export memory data in this format.

The goal of this bachelor thesis is to port our Memory Cities desktop application (built in Unity using C#) to a web-based visualization tool that can be used from within the browser without any manual download or installation needed. To achieve this, Memory Cities should be ported to JavaScript, specifically using the three.js 3D library (https://threejs.org/). All features (time travel, color highlighting, opacity settings, reference visualization, etc.) supported in the desktop application must also be available in the web tool. As a nice-to-have feature, animations could be used to fluently transform from one point in time to another. Since the tool will be extended in the future, readability and a clean code structure is important, as well as clean documentation (i.e., documented method headers).

Modalities:

The progress of the project should be discussed at least every three weeks with the advisor. A time schedule and a milestone plan must be set up within the first 3 weeks and discussed with the advisor and the supervisor. It should be continuously refined and monitored to make sure that the thesis will be completed in time. The final version of the thesis must be submitted not later than 30.09.2022.