Does Asynchronous Programming in JavaScript

affect Software Quality?

A Repository Mining Study on GitHub

How are different Asynchronous Programming Constructs in JavaScript related to Software Quality?

A Repository Mining Study on GitHub

BSc Thesis

**Context & Motivation**

In 1995, JavaScript [1] was developed and introduced for small client-side tasks in the browser. JavaScript spread rapidly with the growth of the internet and is now the most widely used programming language according to a Stackoverflow survey [2]. Reasons for this include its versatility, flexibility, and ease of use. TypeScript [3], a superset of JavaScript, is also becoming more popular. The main difference is its static type system and its class-based object orientation [4]. However, there is a lack of sufficient empirical evidence to support the claim that TypeScript leads to better software quality than JavaScript. The difference between dynamically typed and statically typed languages has been studied occasionally, but results are not conclusive. While some studies claim that statically typed languages have a positive impact on code quality [5], others claim the opposite [6]. There is a lack of studies that directly compare JavaScript and TypeScript projects in terms of software quality on a larger scale.

However, there is still a lack of sufficient empirical evidence for the claim that using functional programming has significant impact on software quality in comparison to other paradigms like procedural or object-oriented programming [6], especially since many other language prejudices could not be supported by studies in the past [7].

The evolution of Web 2.0 technologies makes web applications prevalent in various platforms including mobile devices and smart TVs. While one of the driving technologies of web applications is JavaScript, the extremely dynamic features of JavaScript make it very difficult to define and detect errors in JavaScript applications. The problem becomes more important and complicated for JavaScript web applications which may lead to severe security vulnerabilities.

**Objectives and Tasks**

The goal of this study is therefore to empirically analyze a large set of JavaScript and TypeScript applications. With the help of this data collection, comparative insights into the software quality of the two programming languages should be possible. Additionally, potential influencing factors on the analyzed properties in TypeScript projects should be identified. The quality aspects to be analyzed and the detailed research questions will be defined by the student.

The goal of this study is therefore to empirically analyze projects using functional programming languages and to compare them to projects with non-functional languages. The comparison should provide insights into a potential influence of the functional programming paradigm on software qualities like functional correctness or maintainability. The concrete quality aspects to be analyzed as well as more detailed research questions should be defined by the student.

**Methods**

The research should be conducted as a mining software repository (MSR) study [7][8] using a large number of open-source projects, e.g. on GitHub [9]. Data collection should be conducted with appropriate tools (e.g. static analysis tools) and automated as much as possible to achieve the best possible reproducibility. For the analysis, suitable techniques could be hypothesis testing, correlation, or regression. The detailed study design will be created by the student.

The research will be conducted as a mining software repository (MSR) study [9][10] using a large number of open-source projects on GitHub [11]. Data collection should fully rely on tool support (e.g. static analysis tools) and automation to increase reproducibility. For the analysis, suitable techniques could be hypothesis testing, correlation, or regression. The detailed study design should be created by the student.

**References**