**TypeScript**

* Teknologien forklaret,
* Formål med teknologien
* Anvedelsesområder (eks. design, styling, scripting, animationer osv.)
* Eventuelle forbedringer eller ønsker til teknologien

What is TypeScript?

TypeScript is a free and open-source high-level programming language developed by Microsoft that adds static typing with optional type annotations to JavaScript. It is designed for the development of large applications and transpiles to JavaScript. *\*Wikipedia*

TypeScript is a programming language that adds extra functionality to JavaScript.

TypeScript allows specifying the types of data being passed around within the code, and has the ability to report errors when the types don't match.

For example, TypeScript will report an error when passing a string into a function that expects a number. JavaScript will not.

TypeScript uses compile time type checking. Which means it checks if the specified types match before running the code, not while running the code.

Common way to use TypeScript is to use the official TypeScript compiler, which transpiles TypeScript code into JavaScript. Visual Studio Code, have built-in TypeScript support and can show errors as you write code.  
One also needs to instal Node.js

*For my project, I wrote firstly in TypeScript and then compiled to JavaScript. But due to the nature of the code, it worked only with Vite, which I made a new repository for it. I will understand better why it didn’t work without Vite when I’ll learn more how to use TypeScript properly.*

Purpose of TypeScript

The main goal of TypeScript is to provide strong typing, which helps developers in writing code that others can easily understand. It also provides features like classes and modules.

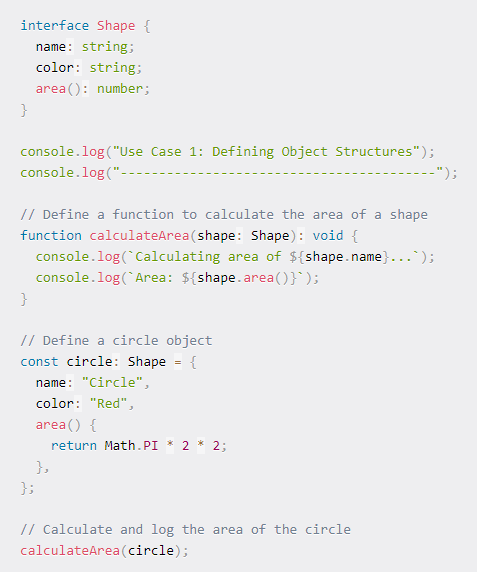
It adds static typings to the language which helps reduce the number of bugs and issues that are created.

TypeScript Interfaces:

Interface is a syntactical contract that defines the expected structure of an object. It provides a way to describe the shape of objects, including their properties and methods, without implementing any functionality. Interfaces solely focus on the structure and type-checking aspects, allowing for better code understanding and validation during development.

Consists of:

* Interface: Keyword used to define an interface.
* InterfaceName: Name of the interface following TypeScript naming conventions.
* property1, property2: Properties of the interface.
* type: TypeScript type annotation defining the type of each property.



In this use case, we define an interface called Shape to represent the structure of geometric shapes. The Shape interface contains properties name and color, both of type string, and a method area() that returns a number. We then define a circle object that adheres to the Shape interface, specifying its properties and implementing the area() method to calculate its area.

Functions:

GETTERS & SETTERS

Getter methods allow you to retrieve the value of a property, while setter methods enable you to modify the value of a property with certain validations or actions.

TypeScript Functions:

In TypeScript, a function type is a powerful feature that allows developers to define the signature of a function, specifying the types of its parameters and return value. This ensures that functions are called with the correct number and type of arguments and that the expected data type is produced. Function types are crucial in maintaining type safety across the application, making the code predictable and less prone to runtime errors.