TypeScript is a strongly-typed, superset of JavaScript that was developed by Microsoft. It adds static type-checking and other features to JavaScript, aiming to improve code quality and developer productivity. Here's an overview of TypeScript, its features, and benefits compared to JavaScript:

Basic Overview:
- **Superset of JavaScript**: TypeScript includes all of JavaScript's features and syntax, allowing developers to use JavaScript code directly within TypeScript files. Existing JavaScript code is valid in TypeScript.
- **Static Typing**: TypeScript adds a type system on top of JavaScript, allowing developers to specify types for variables, functions, and parameters.
- **Compiled Language**: TypeScript code is compiled into JavaScript. This means the final code that runs in the browser or on the server is still JavaScript.
- **ES6 Features and Beyond**: TypeScript supports many modern JavaScript features (such as classes, modules, and arrow functions) as well as experimental or proposed features that might be standardized in the future.

Features:
1. **Type Annotations**: TypeScript allows you to explicitly specify the types of variables, function arguments, and return values. This helps catch errors at compile time.
2. **Interfaces**: TypeScript interfaces allow you to define the structure of objects. Interfaces help enforce consistency in the codebase.
3. **Classes**: TypeScript fully supports JavaScript's ES6 classes and also extends them with additional features like access modifiers (public, private, and protected) and parameter properties.
4. **Generics**: Generics allow you to write functions and classes that work with a variety of types, providing flexibility and type safety.
5. **Type Inference**: In many cases, TypeScript can automatically infer the types of variables based on their usage, reducing the need for explicit type annotations.
6. **Advanced Type System**: TypeScript's type system includes union types, intersection types, mapped types, and more, providing a rich set of tools for expressing complex type relationships.
7. **Tooling and IDE Support**: TypeScript's rich type system enables powerful tooling features such as auto-completion, code navigation, and refactoring support in modern IDEs.

Benefits Compared to JavaScript:
1. **Error Prevention**: The type system helps catch type-related errors during development, reducing runtime errors and improving code quality.
2. **Enhanced Readability**: Type annotations provide additional context about the code, making it easier to understand and maintain.
3. **Refactoring**: With the type system in place, refactoring code (e.g., renaming variables and functions) becomes safer and more straightforward.
4. **Interoperability**: TypeScript is designed to work seamlessly with existing JavaScript code and libraries, allowing you to gradually introduce TypeScript into a JavaScript project.

5. **Community and Ecosystem**: TypeScript has a growing community and ecosystem of libraries and tools that support development.