Test-Driven Development [TDD] is a software development methodology that emphasizes writing automated tests before developing the actual code. This approach ensures that the codebase meets specified requirements and facilitates continuous testing throughout the development process.​

TDD involves a repetitive cycle of three primary steps [1]:​

1. Write a Failing Test: Develop a test case that defines a desired improvement or new function, which initially fails since the functionality isn't implemented yet.​
2. Implement the Code: Write the minimal amount of code necessary to make the test pass.​
3. Refactor the Code: Optimize and clean up the code while ensuring that all tests continue to pass.​

This cycle is repeated for each new feature or bug fix, promoting incremental development and continuous validation of the codebase. ​

Advantages of TDD include improved code quality; by writing tests first, developers clarify the requirements and design, leading to cleaner and more reliable code. Additionally, early bug detection; since tests are written before the code, issues can be identified and addressed early in the development process, reducing the cost and effort of fixing them later. ​Finally, facilitates refactoring; The presence of a comprehensive test suite allows developers to refactor code confidently, knowing that existing functionality is preserved.

Disadvantages of TDD include initial time investment; writing tests before code can slow down initial development, as it requires additional time and effort upfront. Additionally, the learning curve may be steep. Teams new to TDD may require training and adaptation to integrate TDD practices effectively. ​Finally, the potential for incomplete testing: If tests are not comprehensive, there's a risk of missing edge cases or complex scenarios, leading to a false sense of security [2]. ​

TDD was developed by Kent Beck in the late 1990s as part of the Extreme Programming methodology [3]. The primary motivation was to encourage simple design and instill confidence in the software development process. By writing tests first, developers can focus on the requirements and design, leading to more maintainable and robust code. ​

Sources:

1. <https://martinfowler.com/bliki/TestDrivenDevelopment.html>
2. <https://medium.com/%40dmautomationqa/the-pros-and-cons-of-using-test-driven-development-tdd-in-software-development-6fd41f50e995>
3. https://www.educative.io/blog/test-driven-development