When you do a code review, what do you do first? Do you dive into the commits, the merge request or the pull request and review the changes? Or do you verify that the code actually works? When I started doing code reviews, I assumed that the developer committed a working solution so I dove right into the code. After doing many code reviews over the years, I realized that this assumption can be wrong. Therefore, as reviewers, I believe step number one needs to answer this question, does it work?

So, how do we verify that the code we are reviewing actually works?

Start with reviewing the story or bug report and understanding what the code is supposed to do. A story should provide enough detail (sometimes called acceptance criteria) to allow one to verify working code in a straightforward manner. A bug report should include reproduction steps, observed behavior and expected behavior. If there is any ambiguity or confusion after reviewing the story or bug report, seek clarity from those who created the story or bug report.

The next step is pulling the code from source control and building it locally. The code should build cleanly with no errors, no warnings and no test failures. Obviously, build errors are unacceptable and will stop a review immediately. The errors I see are usually a result of not configuring or committing build dependencies correctly. In my experiences, it is rare to see a build error resulting directly from the committed code itself. Warnings should also stop a review immediately. Warnings can point to potentially problematic code that could lead to bugs. Test failures should also stop a review immediately. The developer might state that the tests pass on his or her machine. If this is the case, a failing test on another machine suggests that something was missed when the code was committed. Communicate any build errors, warnings or test failures immediately to the developer.

After the code builds cleanly, run and test the code. When the code is started, there should be no unexpected start up behavior (examples: unexpected errors or warnings in the log or the UI). If the code does not start correctly, this could point to missed runtime dependencies or missed configuration changes. Unexpected start up behavior should stop the review and be communicated to the developer. Assuming normal start up behavior, verify that the code does what the story states it should do or that it corrects the behavior observed in the bug report. Do enough testing to reach the conclusion, "yes, this works" or "no, it doesn't".

What if it is not practical for the reviewer to build and test the code locally? In this case, the developer should demonstrate to the reviewer that the code builds cleanly as described above and the code executes correctly as described above.