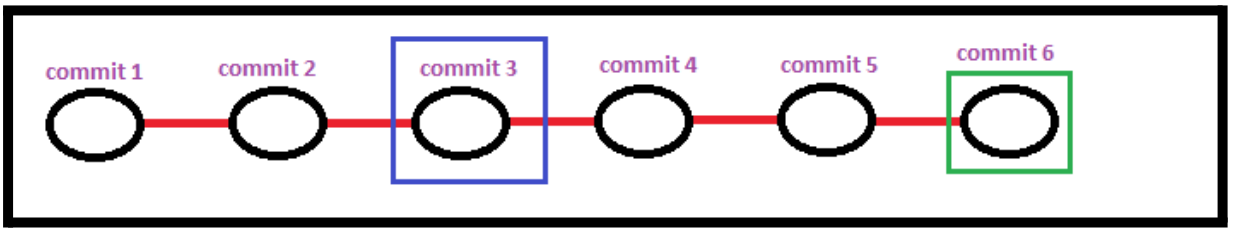
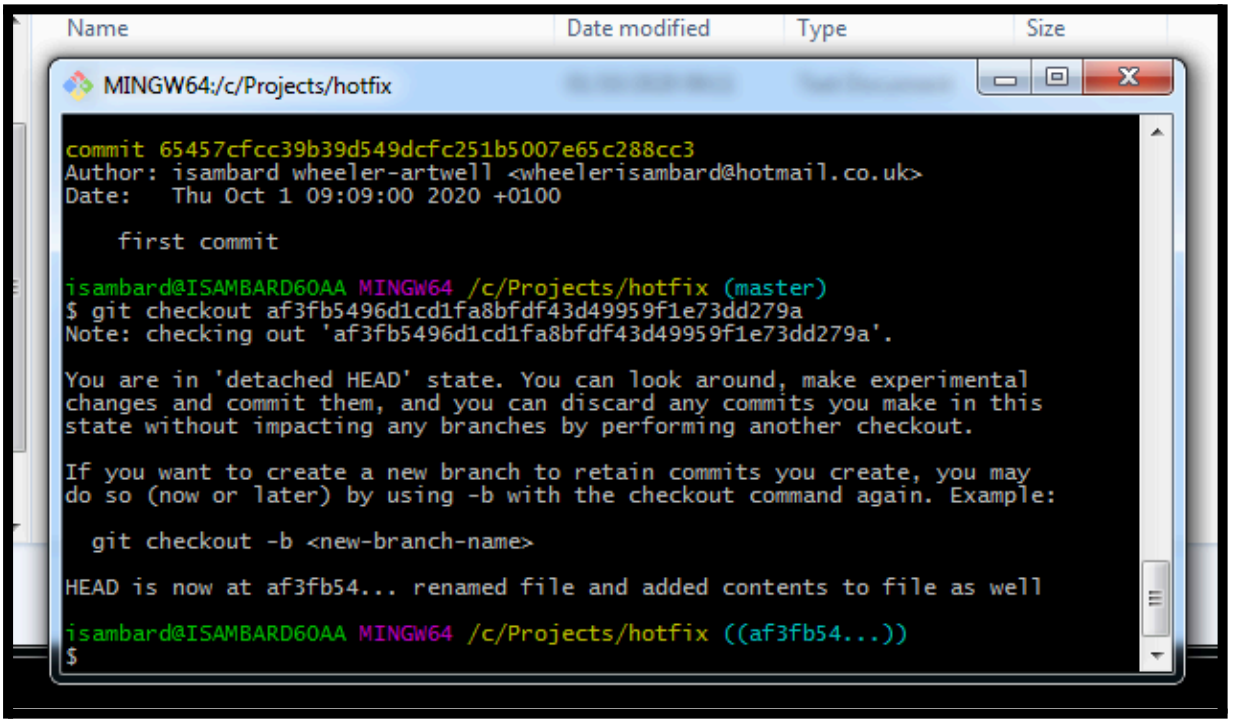
Git – hotfixing

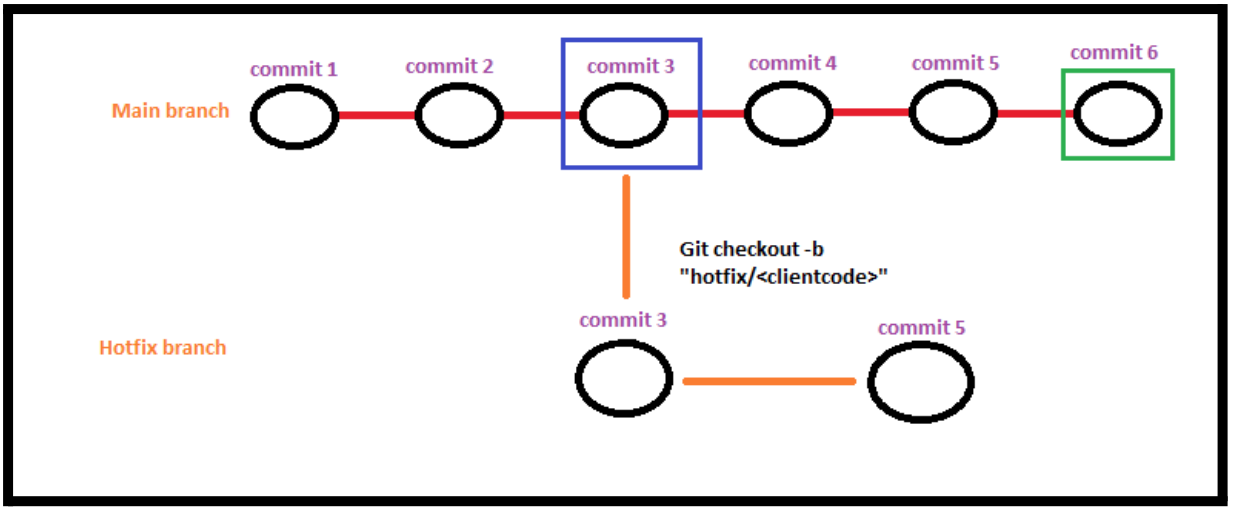
What is HotFixing

Hotfixing, occurs in software development when there is a bug on the client's live code base which they need to address quickly. They are pretty much 100% happy with the software. However, there may be one problem or feature that is not working correctly. With hotfixing we can quickly amend the end user's code base.
There are two avenues of thought in this case. As the picture depicts below. The software for the client was deployed at commit 3. Although development has progressed and the team has progressed the code on the main branch to commit 6.

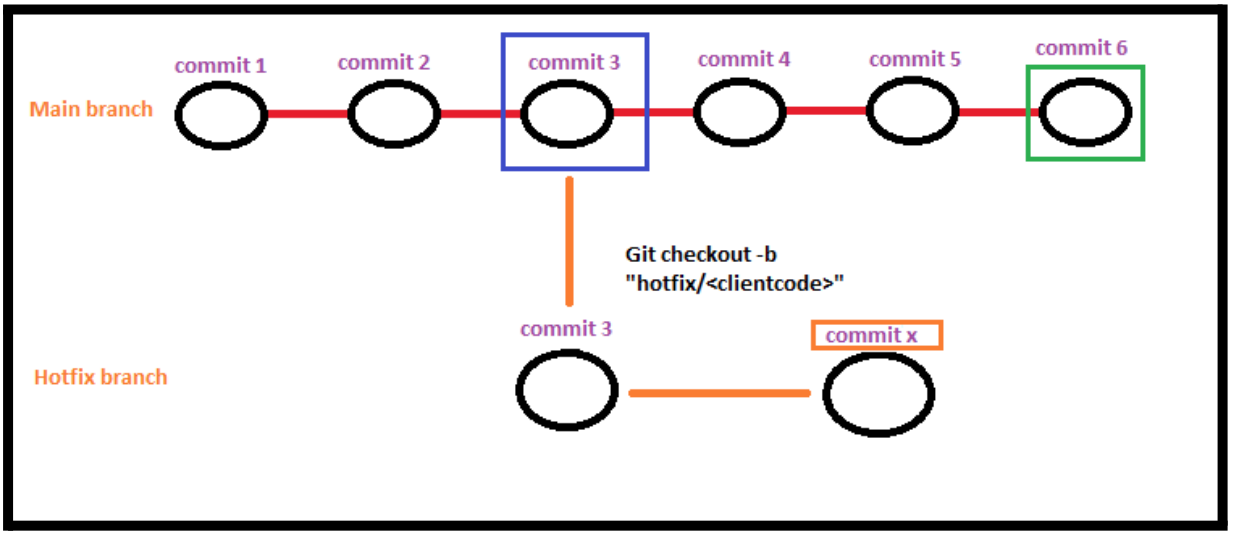


The first task is to do a git checkout commit 3 into the terminal. This will get our code to the same level as the code for the client. When we do this we will get a detached HEAD state in the terminal. What this means is that we have checkout out to commit 3 but git is still pointing to commit 6. At this moment we can create a branch called Hotfix/<clientcode>





In the image above, when we checkout the code on commit 3 we then apply one more commit. Commit 5. Let's argue that commit 5 contains the fix that we require. Then in our build machine and our server we build the Hotfixing branch and deploy it the client's live environment. Which means that the Hotfixing branch is now live as supposed to the main branch. This in summary, is one avenue – the most simple avenue in this example.
The second avenue that could be presented to us. Is the situation below. As we have highlighted below the commit x is highlighted in a square box. This represents a situation in which the commit might not exist in the main branch.



When this happens, then we need to be sure to add commit x into our main branch as well. The reason being is even though the hotfix branch will go live. When we go back to deploy the main branch then we need to have commit x in the branch as well otherwise the bug will still be there.