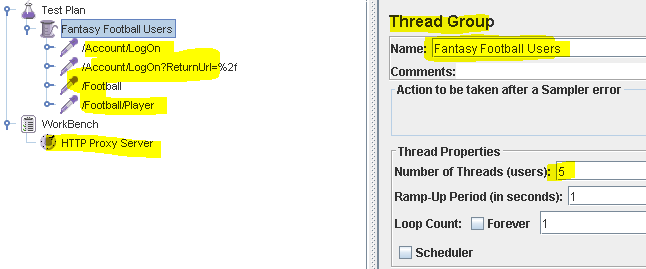
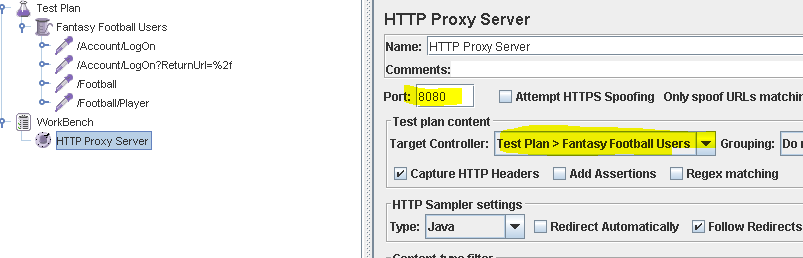
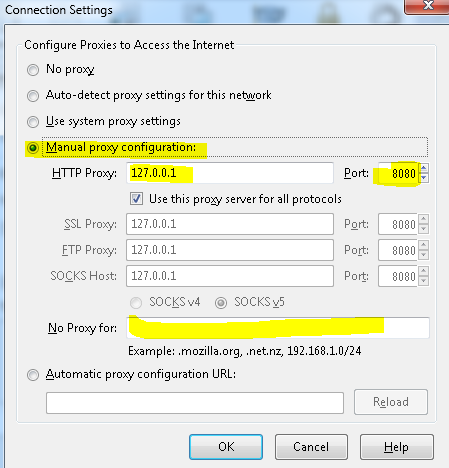
Setup JMeter to Record (although start with 1 user)

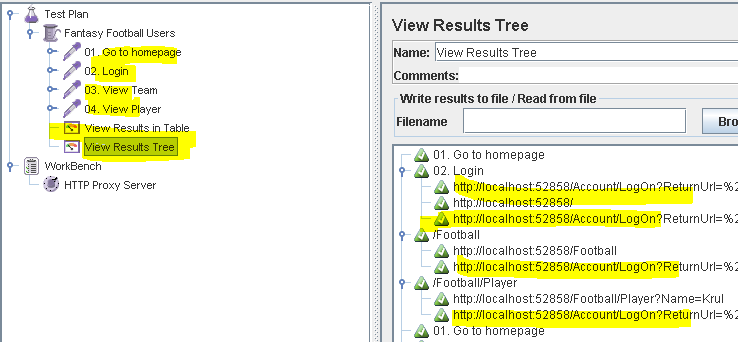




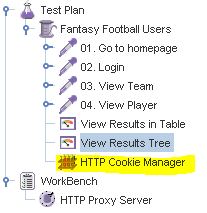
Setup Firefox to proxy requests through jmeter



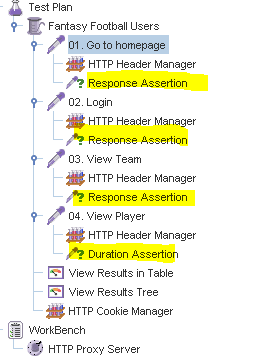


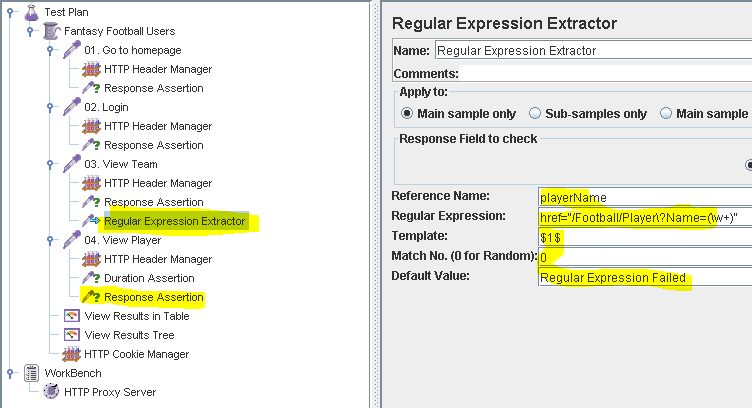


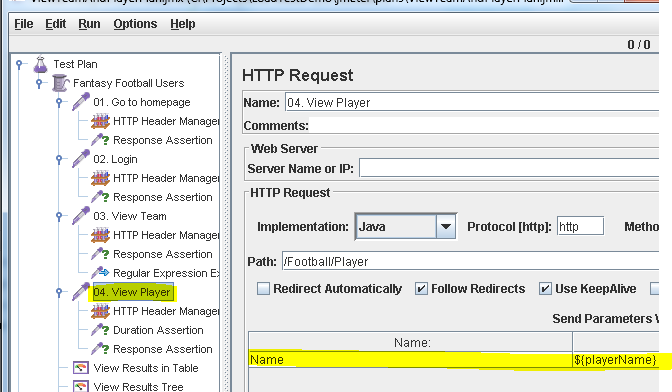
What’s going on? Why redirect to log-in for each step?



Add some response and duration assertions

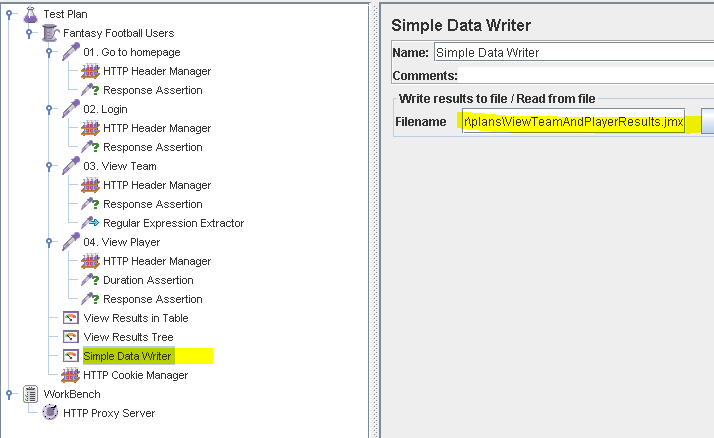


What about clicking on a random player in the team? 

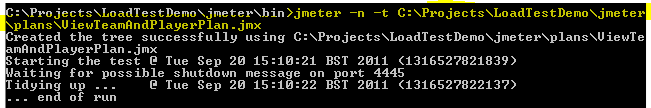




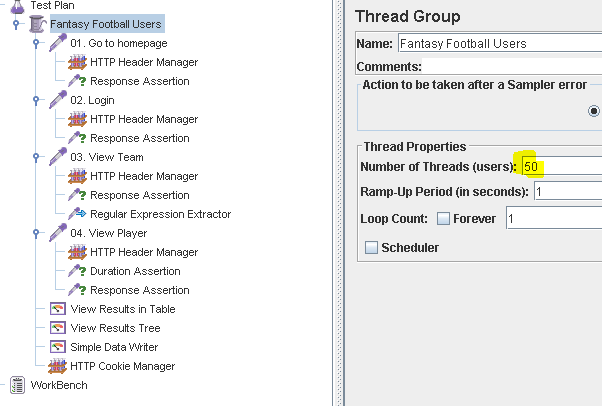
Save Results?



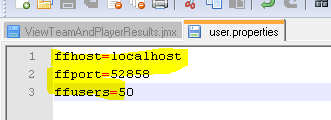
To integrate with CI we’ll need to be able to run this chit headless!



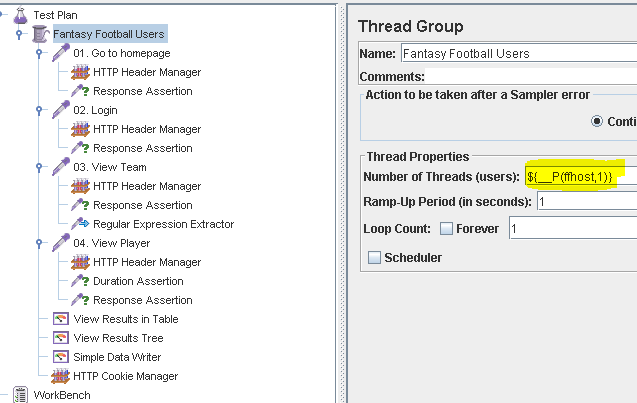
Check the report is still being generated! Boom! Let’s ramp it up to 50 users…



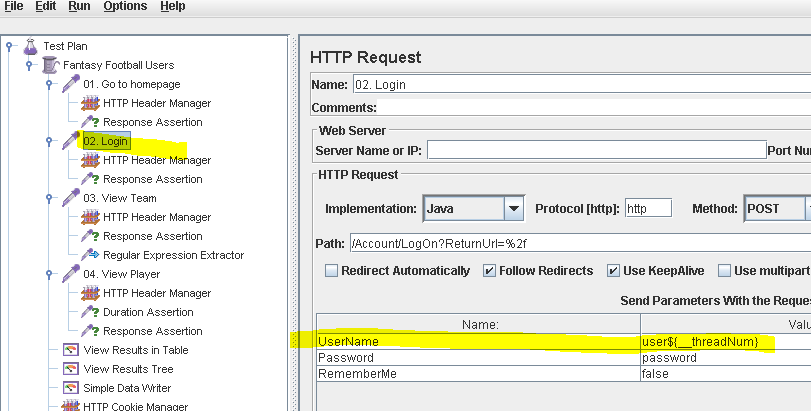
In fact let’s make the number of users, the host and the port parameterizable from the command line. user.properties in the jmeter bin folder.



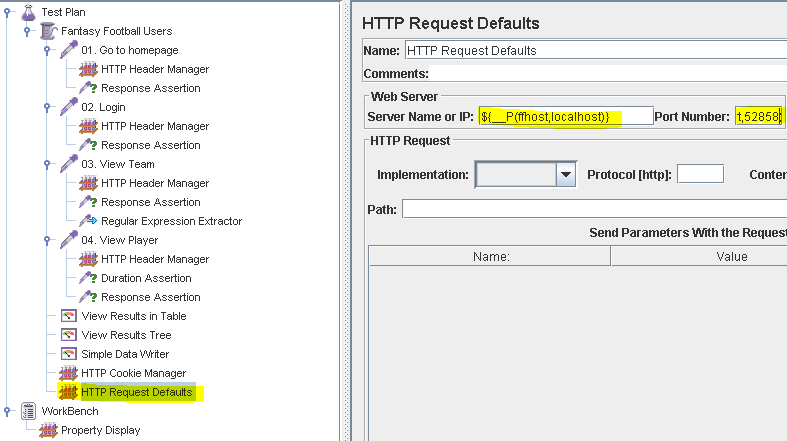
Users



And also make them different users!

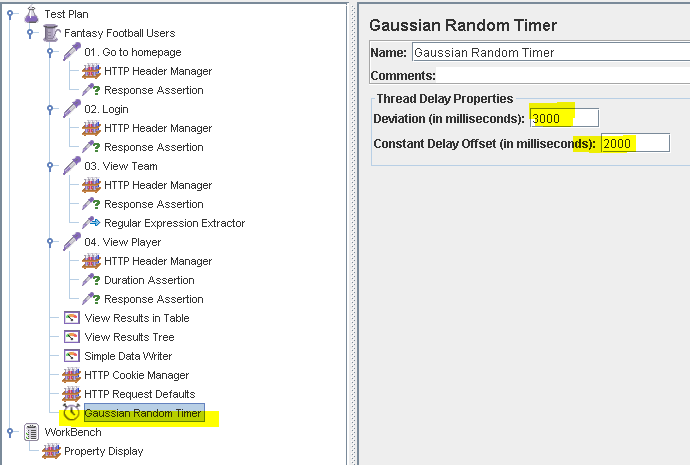


Host and Port

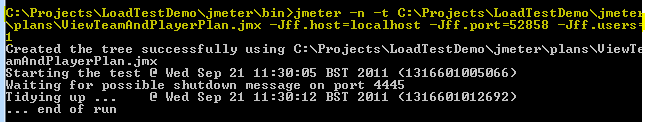


Gotta remove old Host & Ports from existing requests too…

Add user think time into the equation:

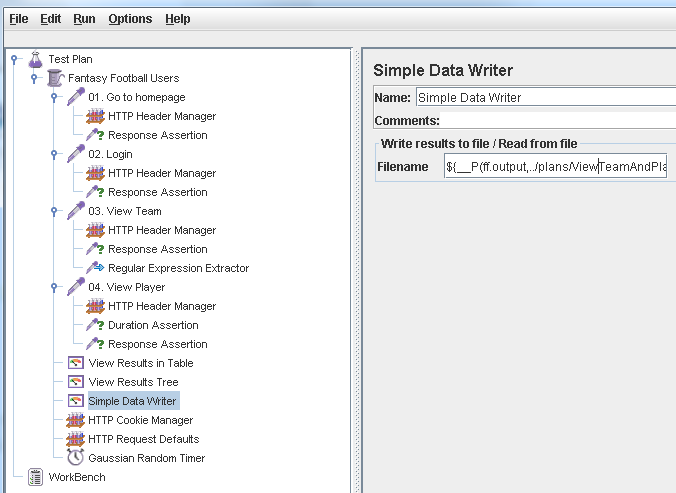


Let’s run that from the command line!



By now you’ve probably noticed that all those boxes in the great GUI are dynamically parsed. Cool.

We’ll need to parameterise one more thing before we can move it into CI, where the output file gets stashed.



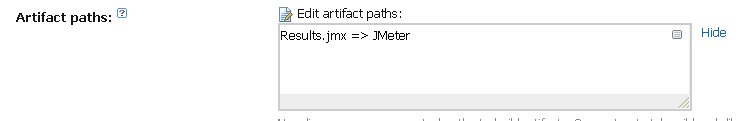
${\_\_P(ff.output,../plans/ViewTeamAndPlayerResults.jmx)}

You could include the date in the filename or whatever, however we’ll lean on artefact-ing in Team City instead.

Next we need a new msbuild project to run our script and capture the output.



Wire up artifacts



Run a build and we should see the Results.jmx artefact created and stored for us…

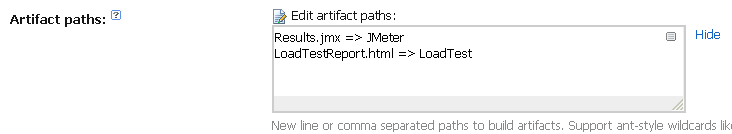
Would be nice to do something a little more interesting with that information….

[Look at my custom log parser console application]

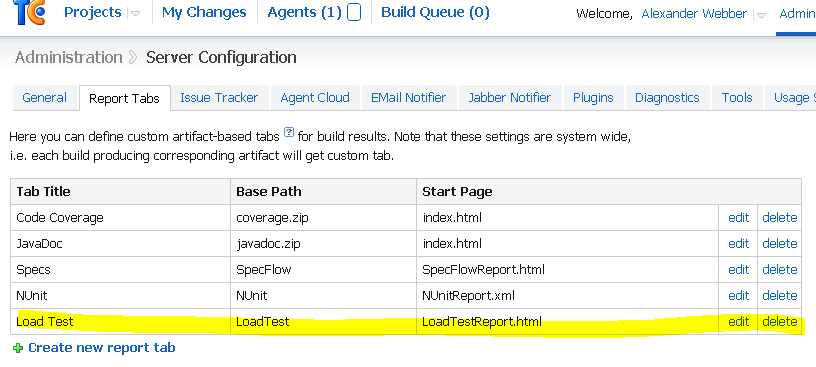
Look at simple MSBuild task that uses it..



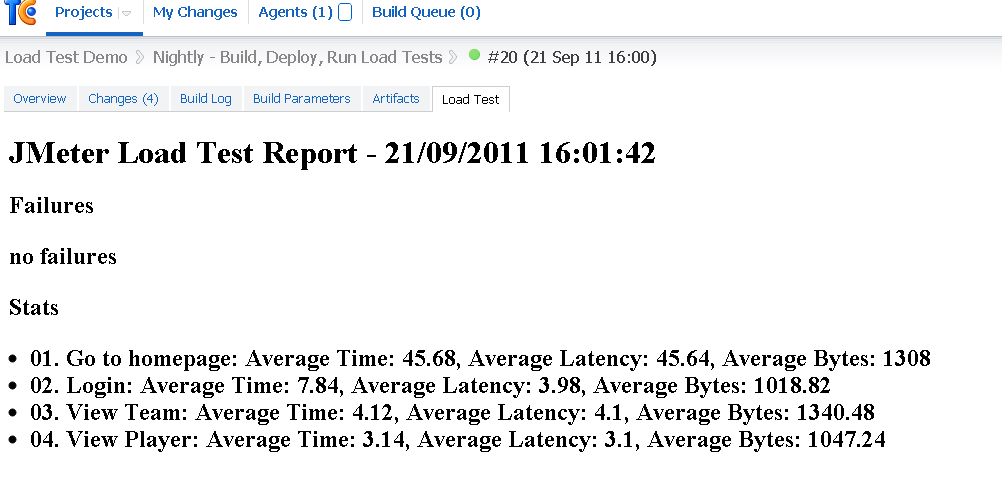
Let’s hook this up to a report tab in Team City. So another Artifact:



And the global report tab itself:

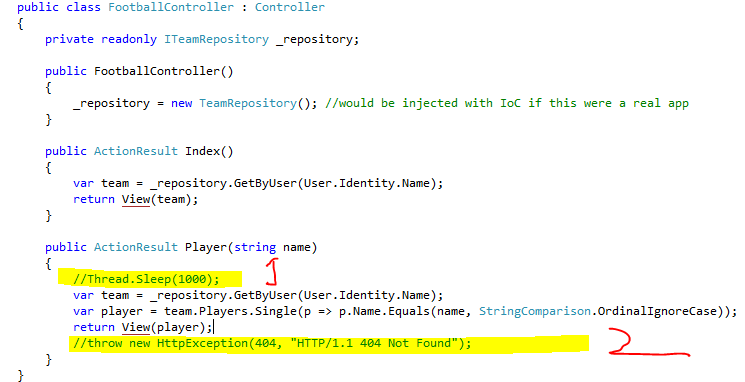


Let’s run this and see what shows in the tab.



Boom!

Add a thread.sleep to simulate heavy operation (lol) and then a 404 to represent an unexpected response.



Check-in and run the build again.