



Magazine

# OpenSta – OpenSource dla Web Load, HTTP Stress & Performance testing

---

*Autor:* Łukasz Smolarski



## *O autorze:*

Łukasz Smolarski jest absolwentem Wyższej Szkoły Biznesu-National Louis University w Nowym Sączu na kierunku Informatyka oraz Akademii Leona Koźmińskiego w Warszawie na kierunku Zarządzanie. Podczas studiów wygrał stypendium dla liderów ufundowane przez GE Foundation. Obecnie pracuje w firmie Gtech Polska na stanowisku Quality Software Engineer, gdzie odpowiada za automatyzację testów. W 2007 zdał ISTQB Foundation Level, a w 2010 został Certyfikowanym Specjalistą HP - Mercury Quality Center oraz Mercury QuickTestPro. Członek SJSI.

## **Intermediate**

Level

## 2

Magazine Number

## Testowanie oprogramowania

Section in the magazine

### Wprowadzenie

Każdego dnia w naszej pracy spotykamy się z powtarzaniem wszelkiego rodzaju testów, co zabiera nam mnóstwo czasu. Nieraz jakże ciężka jest to praca, gdy musimy zrobić coś kilkanaście lub nawet kilkaset razy. Jednym z programów, które mogą w tym pomóc i który chciałbym opisać jest OpenSta, narzędzie służące do mierzenia wydajności oprogramowania, ale także do innych celów jak np. możliwość zautomatyzowanie pewnych akcji.

OpenSta jest ciągle rozwijanym narzędziem darmowym, który możemy pobrać ze strony <http://opensta.org>. Ponadto jeśli mamy jakiś problem lub pytanie możemy skorzystać z forum użytkowników tego programu, które również znajduje się na podanej stronie.

### Pierwsze kroki

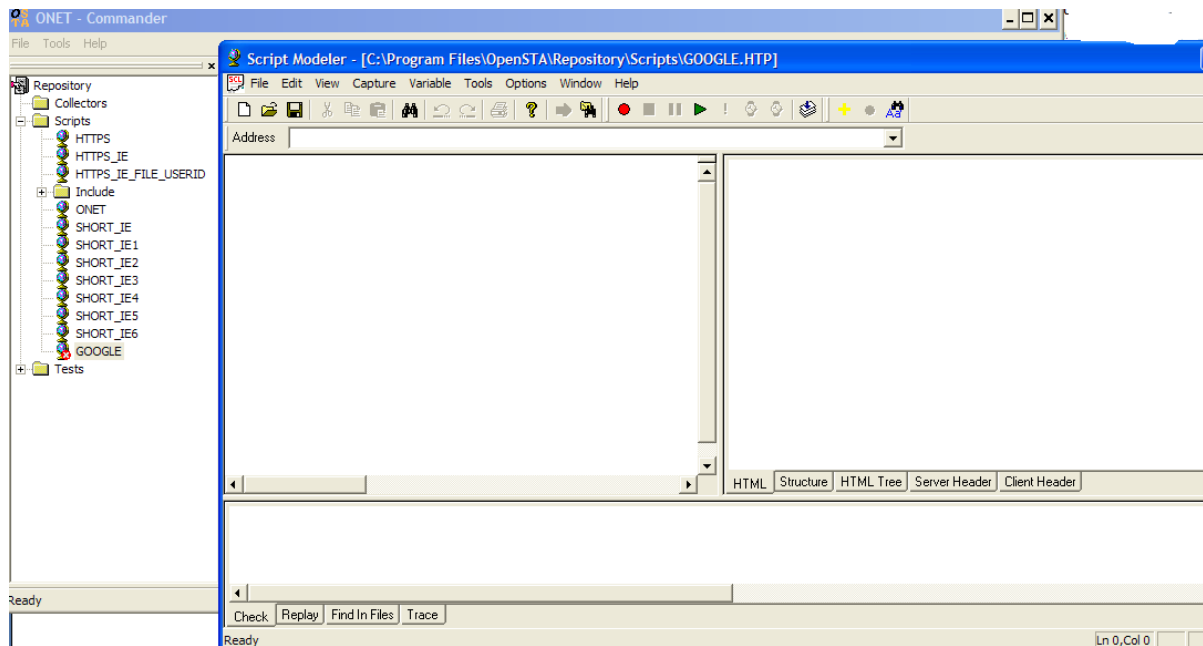
Po instalacji uruchamiamy OpenSta Commander. Jest to główny ekran programu, który składa się z drzewa będącego naszym repozytorium, w którym będą znajdować się nasze skrypty i testy.

Struktura składa się z trzech katalogów: Collectors, Scripts, Tests. My zajmiemy się katalogiem Scripts, w którym tworzone są skrypty w języku SCL, oraz katalogiem Tests, w którym tworzone są testy korzystające ze skryptów stworzonych wcześniej. Ponadto w górnej części aplikacji mieści się pasek, w którym dostępne są różne opcje - między innymi dość rozbudowany HELP.



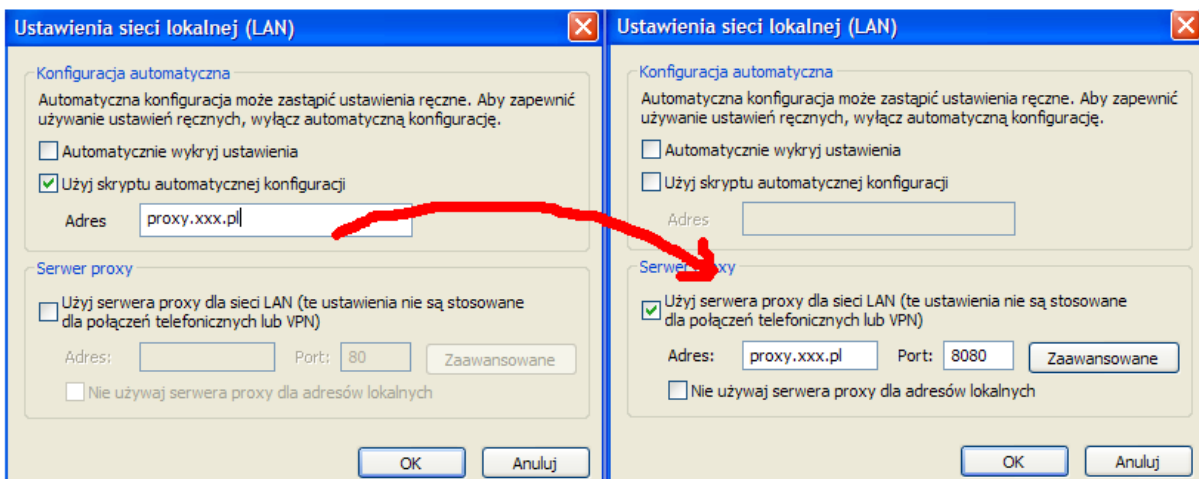
Rysunek 1 Główne okno programu

Aby utworzyć nowy test musimy zacząć od stworzenia skryptu do testowania. W tym celu wybieramy z menu **File ->New Script** . Po stworzeniu klikamy na niego dwa razy i pojawia się nam okno do nagrywania skryptu.



Rysunek 2 Okno do tworzenia skryptów

OpenSta może nagrywać zarówno po http jak i HTTPS, co na przykład odróżnia go od JMetera. Jeśli łącząc się z internetem, korzystamy z serwera Proxy, należy pamiętać o odpowiedniej konfiguracji naszej przeglądarki. Wyboru dokonujemy z menu **Options -> Browser**. Jeśli chcemy się łączyć z

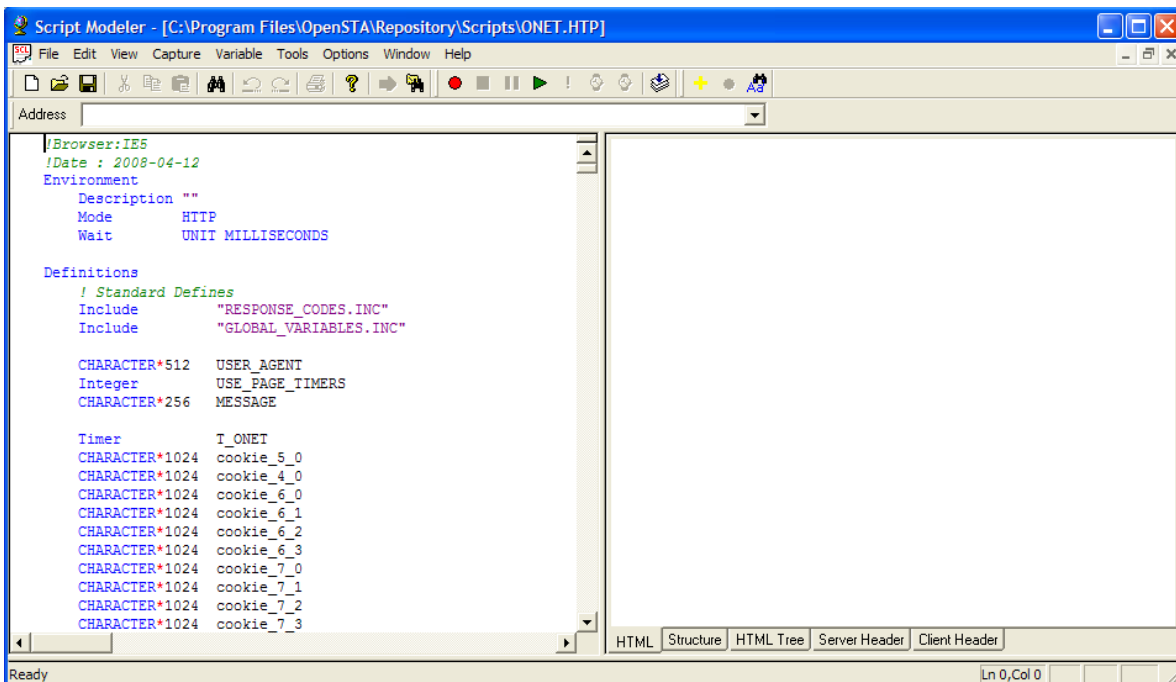


Rysunek 3 Konfiguracja Proxy

odległym serwerem, jest to również możliwe poprzez odpowiednie ustawienia **Options->Gateway**. Niestety tutaj napotykamy na pierwszy problem. Jeśli mamy Serwer Proxy ustawiony w ten sposób OpenSta nie będzie potrafił wystartować . Trzeba przepisać Proxy tak, jak na zamieszczonym obrazku.

Wartą wspomnienia opcją jest niewątpliwie możliwość deklaracji zmiennych z górnego menu **Variable ->Create**, gdyż możemy jeszcze przed nagraniem kodu przygotować zmienne do przechowywania interesujących nas wartości. W celu rozpoczęcia nagrywania klikamy na czerwony przycisk lub wybieramy z menu **Capture->Record**. Otwarta zostaje przeglądarka, w której

wykonujemy zaplanowany scenariusz. Po zakończeniu zamykamy przeglądarkę lub klikamy przycisk Stop .



```
Script Modeler - [C:\Program Files\OpenSTA\Repository\Scripts\ONET.HTP]
File Edit View Capture Variable Tools Options Window Help
Address
!Browser:IE5
!Date : 2008-04-12
Environment
  Description ""
  Mode HTTP
  Wait UNIT MILLISECONDS

Definitions
! Standard Defines
  Include "RESPONSE_CODES.INC"
  Include "GLOBAL_VARIABLES.INC"

CHARACTER*512 USER_AGENT
Integer USE_PAGE_TIMERS
CHARACTER*256 MESSAGE

Timer T_ONET
CHARACTER*1024 cookie_5_0
CHARACTER*1024 cookie_4_0
CHARACTER*1024 cookie_6_0
CHARACTER*1024 cookie_6_1
CHARACTER*1024 cookie_6_2
CHARACTER*1024 cookie_6_3
CHARACTER*1024 cookie_7_0
CHARACTER*1024 cookie_7_1
CHARACTER*1024 cookie_7_2
CHARACTER*1024 cookie_7_3

HTML Structure HTML Tree Server Header Client Header
Ready Ln 0, Col 0
```

Rysunek 4 Kod programu

Po lewej znajduje się kod programu, deklaracje zmiennych, informacje o środowisku oraz inne dane, które OpenSta przechwyił. Kod, jak wspomniałem, jest w języku SCL (Structured Control Language). Jeśli po nagraniu istnieje potrzeba np. wyciągnięcia danych, zrobienia pętli lub innych instrukcji, dopisujemy je bezpośrednio w kodzie. Po skończonej pracy kompilujemy i jeśli nie wystąpiły żadne błędy uruchamiamy skrypt klikając na zielony przycisk „play”.

Język programowania nie należy do najprostszych, jednak jeśli przyjrzymy się z bliska możemy zauważyć pewne zależności jak np. podstawianie danych pod zmienne itp. Na następnym obrazku pokażę fragment kodu, który zmodyfikowałem w celu wyciągnięcia z kodu HTML danych generowanych przez JavaScript, gdyż OpenSta nagrał to na stałe i podczas prób ponownego wykonywania skryptu podstawiał starą wartość niezgodną z aktualnie generowaną przez stronę.

```
"raw/purchaseDrawGame.do?gameId=9101", &
"Accept-Language: pl", &
"Content-Type: application/x-www-form-urlencoded", &
"Connection: Keep-Alive", &
"Content-Length: 1107", &
"Pragma: no-cache", &
"Cookie: "+cookie_2_1+"; "+cookie_1_1+"; "+cookie_10_0+"; "+s_cookie_22_0) &
,BODY "org.apache.struts.taglib.html.TOKEN="+wyjete_dynamiczne_dane+"zboards$5B0$5D." &
"D=QPzboards$5B0$5D.picks$5B1$5D=QPzboards$5B0$5D.picks$5B2$5D=QPzboards$5B0$5D.picks$5B3$5D=QP

Load Response_Info Header on 6 &
Into cookie_6_0 &
,WITH "Set-Cookie,tracking"

Load Response_Info Body on 6 &
Into Interesujaca_mnie_wartosc &
,WITH "HTML(0)/BODY(1)/BODY(2)/TABLE(1)/TBODY(0)/TR(1)/TD(1)/FORM(1)" &
"/INPUT(0):ATTRIBUTE:value(2)"

Log "To moja wartosc " , Interesujaca_mnie_wartosc
```

Rysunek 5 Zmodyfikowany kod programu

Save the active document Ln 2623, Col 53

Jeśli nagrany skrypt nie posiada błędów powinniśmy zobaczyć następujący komentarz :

```
!Browser:IE5
!Date : 2008-04-12
Environment
Description ""
Mode HTTP
Wait UNIT MILLISECONDS

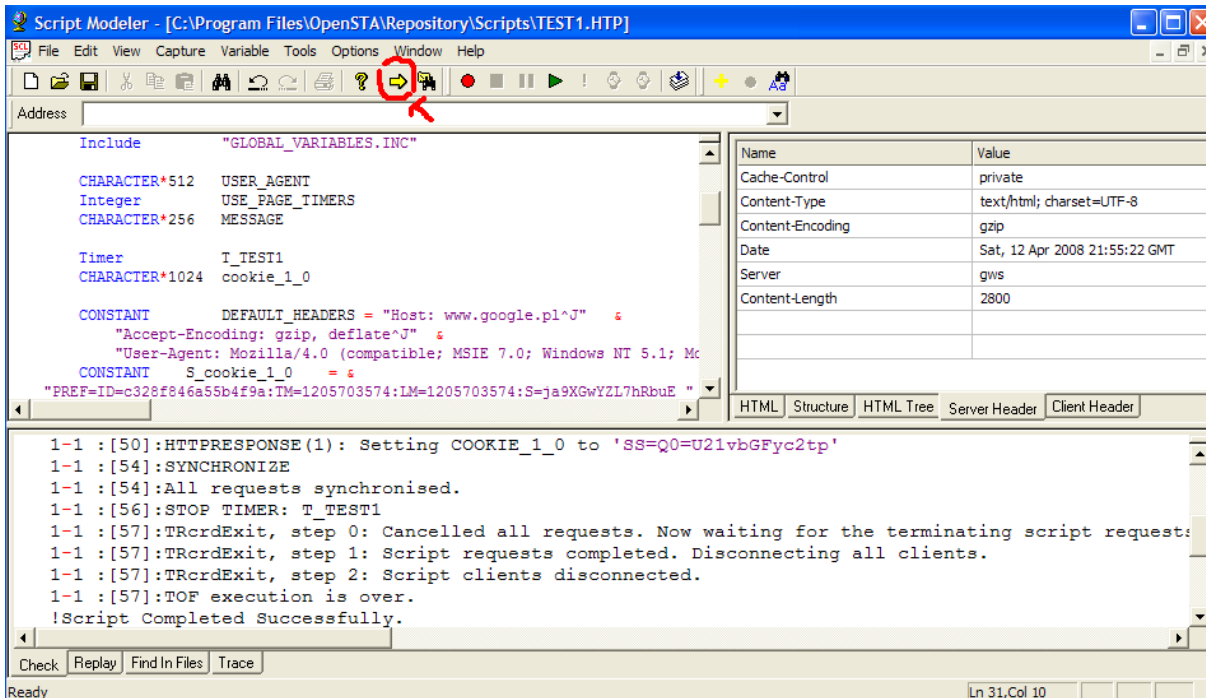
Definitions
! Standard Defines
Include "RESPONSE_CODES.INC"
Include "GLOBAL_VARIABLES.INC"

CHARACTER*512 USER_AGENT
Integer USE_PAGE_TIMERS

1-1 :[26]:SYNCHRONIZE
1-1 :[26]:All requests synchronised.
1-1 :[28]:STOP TIMER: T_TEST
1-1 :[29]:TRcrdExit, step 0: Cancelled all requests. Now waiting for the terminating script request:
1-1 :[29]:TRcrdExit, step 1: Script requests completed. Disconnecting all clients.
1-1 :[29]:TRcrdExit, step 2: Script clients disconnected.
1-1 :[29]:TOF execution is over.
!Script Completed Successfully.
```

Rysunek 6 Kompilacja kodu

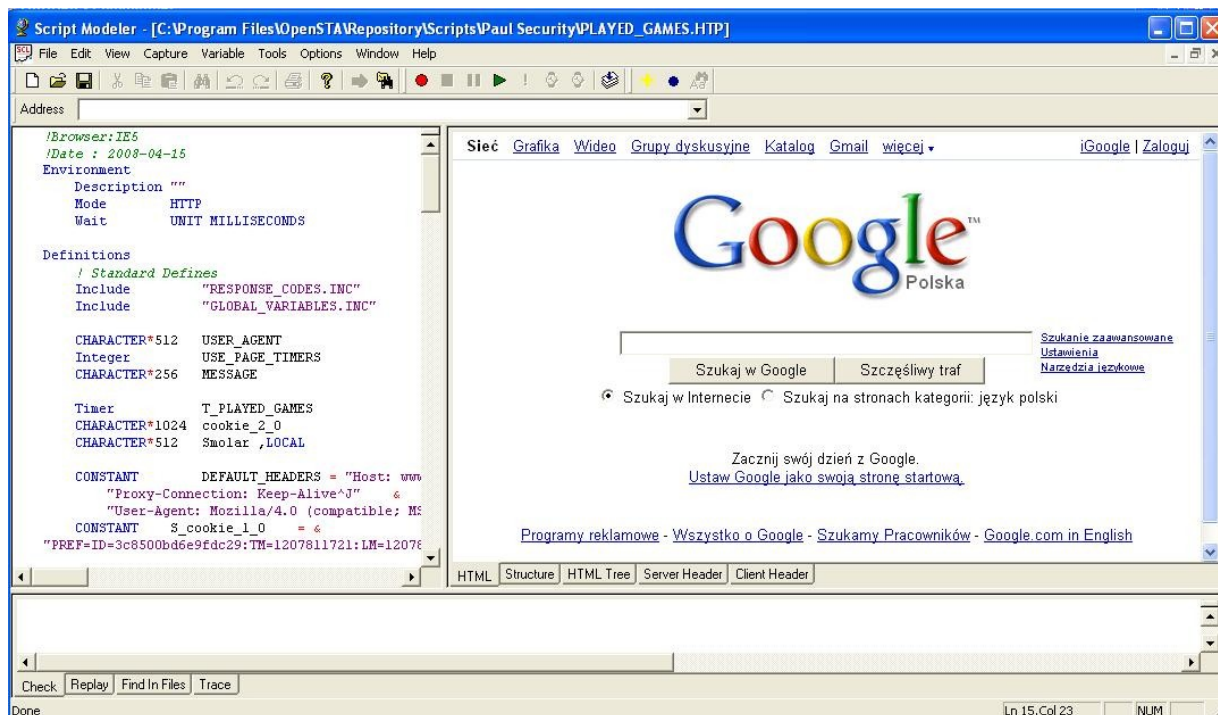
Po kompilacji, możemy znaleźć w kodzie funkcję „Get” najść na nią kursorem i gdy uaktywni się żółta strzałka w górnym menu nacisnąć ją. Zobaczmy stronę w podglądzie w prawej kolumnie.



Rysunek 7 Uruchomienie podglądu strony po prawej stronie

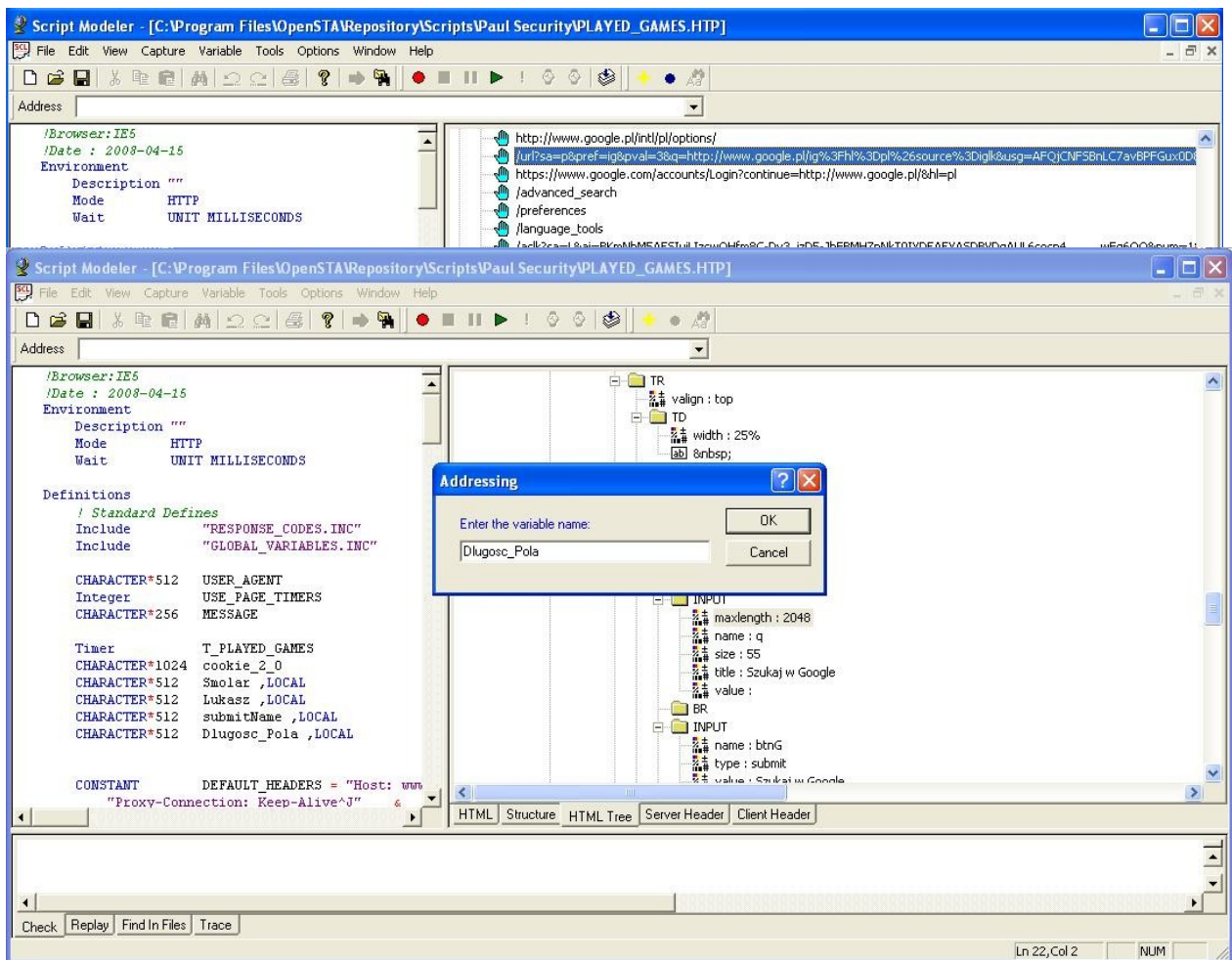
Teraz możemy zobaczyć strukturę HTML, dane o serwerach i inne mogące nam pomóc informacje.

Możemy też przejść do kodu HTML i klikając prawym przyciskiem myszy na interesujący nas kod stworzyć zmienne, które będą nam wyciągać elementy z modułu DOM z kodu (może to nam posłużyć do wyciągania różnych informacji np. tworzonych przez język JavaScript),



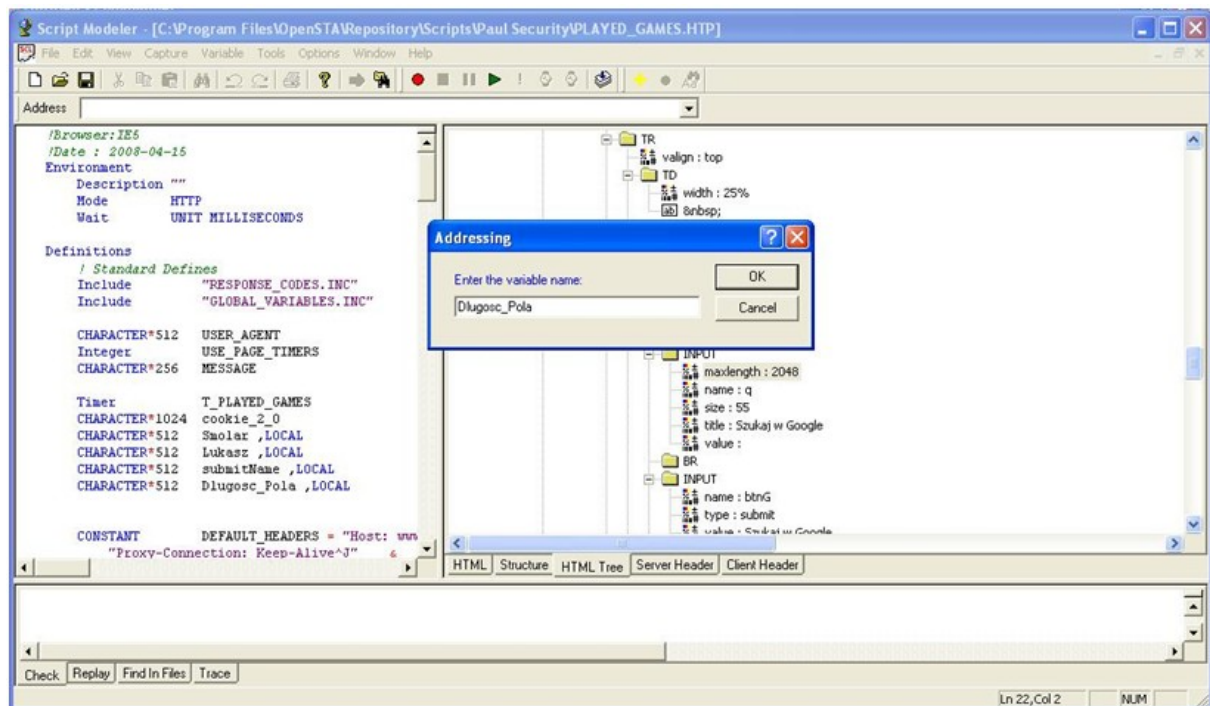
Rysunek 8 Podgląd strony

Klikając na jedną z zakładek podglądu strony możemy zobaczyć całą strukturę strony, elementy oraz wszystkie wartości.



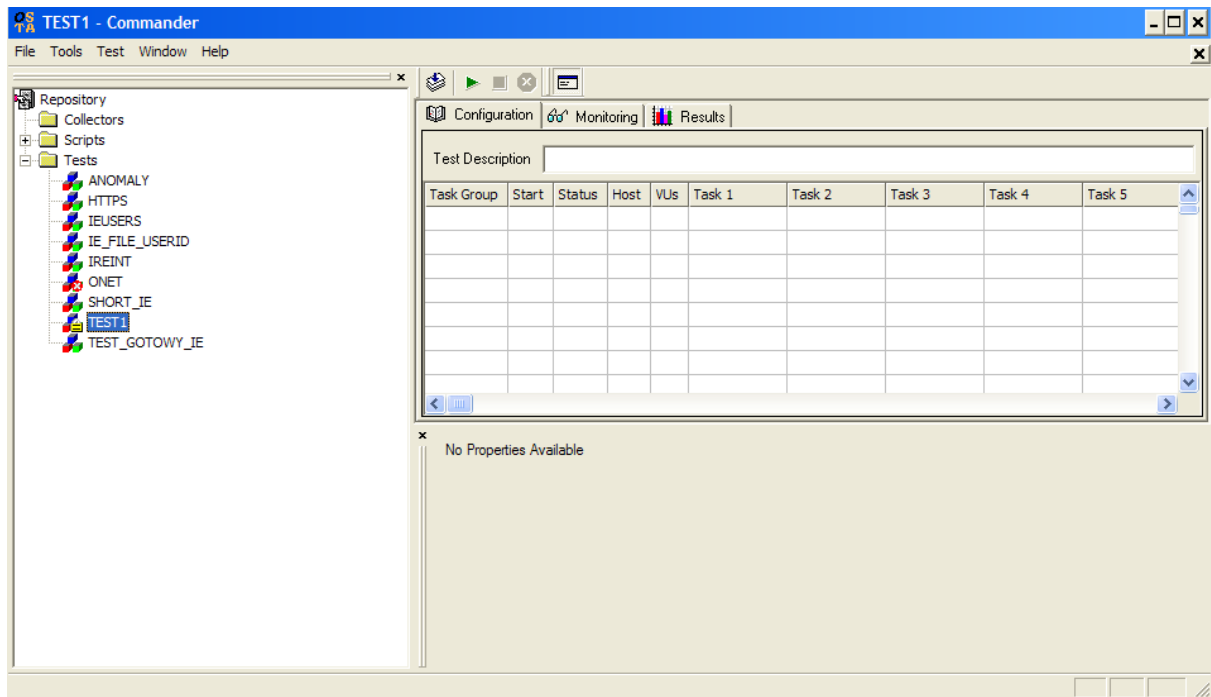
Rysunek 9 Struktura strony

Ponadto klikając na zakładkę HTML Tree możemy znaleźć w kodzie interesującą nas wartość, po czym klikając prawym przyciskiem myszki podstawić ją bezpośrednio pod zmienną do naszego kodu po lewej stronie zakładki.



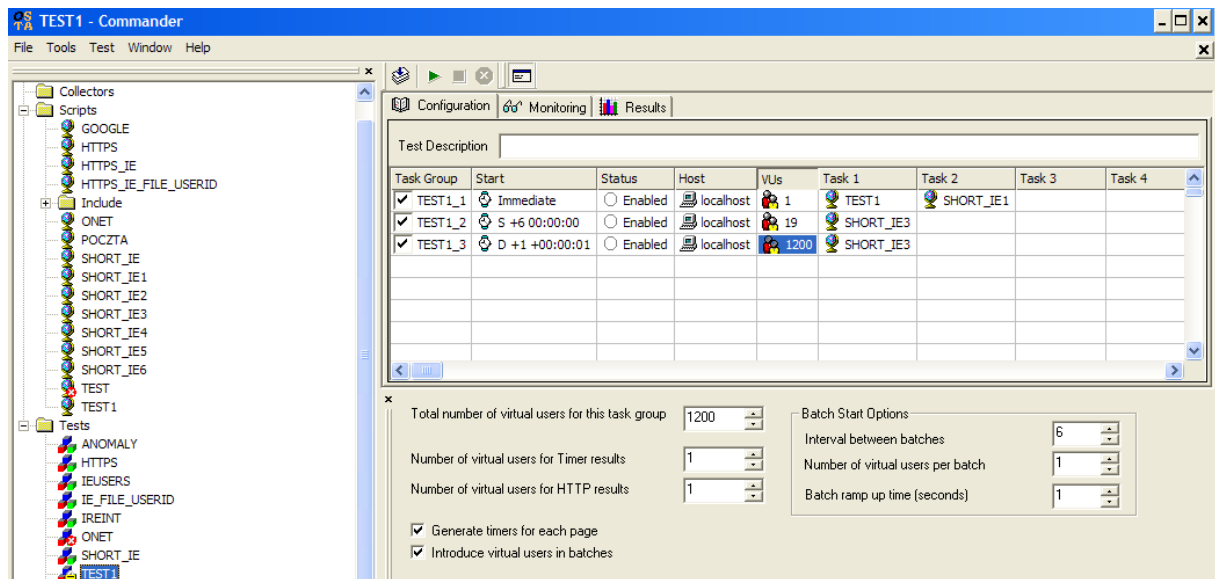
Rysunek 10 Podstawianie wartości z HTML pod zmienną

Teraz możemy zająć się tworzeniem testu. Powracamy do głównego menu programu i klikamy z menu **File-> New Test-> Tests** . Po utworzeniu dwukrotnie klikamy na ikonę testu w drzewie po czym pojawia się nam menu testowe.



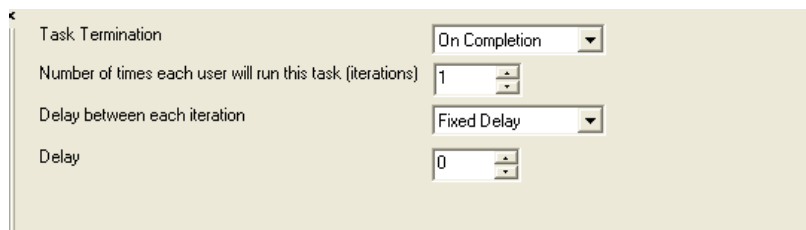
Rysunek 11 Okno do tworzenia testów

Następnie rozwijamy listę ze skryptami, wybieramy stworzony wcześniej skrypt i przeciągamy go do pola **Task**. Pola te mamy podzielone na kolumny w które możemy dodawać kilka skryptów. Oznacza to, że jeśli dodamy w tym samym wierszu - ale różnych kolumnach - kilka testów, wszystkie zostaną wykonane w tym samym czasie, natomiast dodanie skryptów w różnych wierszach stworzy nam scenariusz przejść jedno po drugim.



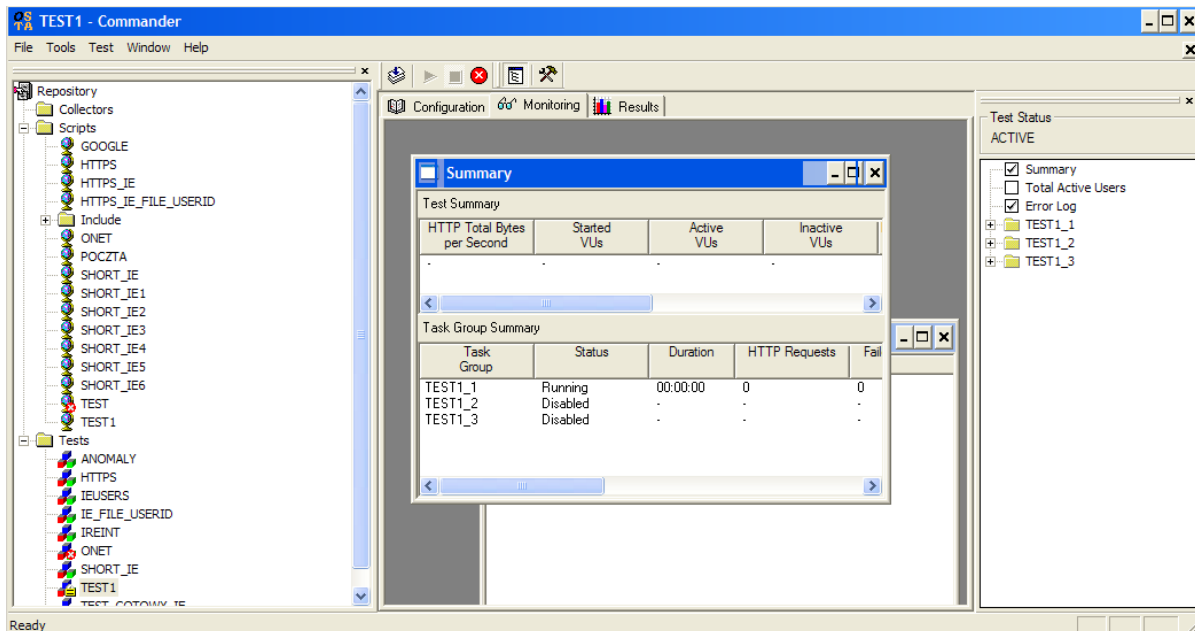
Rysunek 12 Okno do konfiguracji testów

W konfiguracji testów mamy kilka pól, którymi możemy sterować. Pole **Start** służy do ustawiania czasu, kiedy test ma się wykonać (bezpośrednio, opóźniony i zaplanowany). Kolejnym ustawieniem jest ilość wirtualnych użytkowników dla każdego zadania. Użytkowników tych możemy podzielić na różne kategorie np. dajemy całkowitą ilość VU 1200, przydzielamy po jednym użytkowniku dla „Timer Results” i dla „HTTP results”.



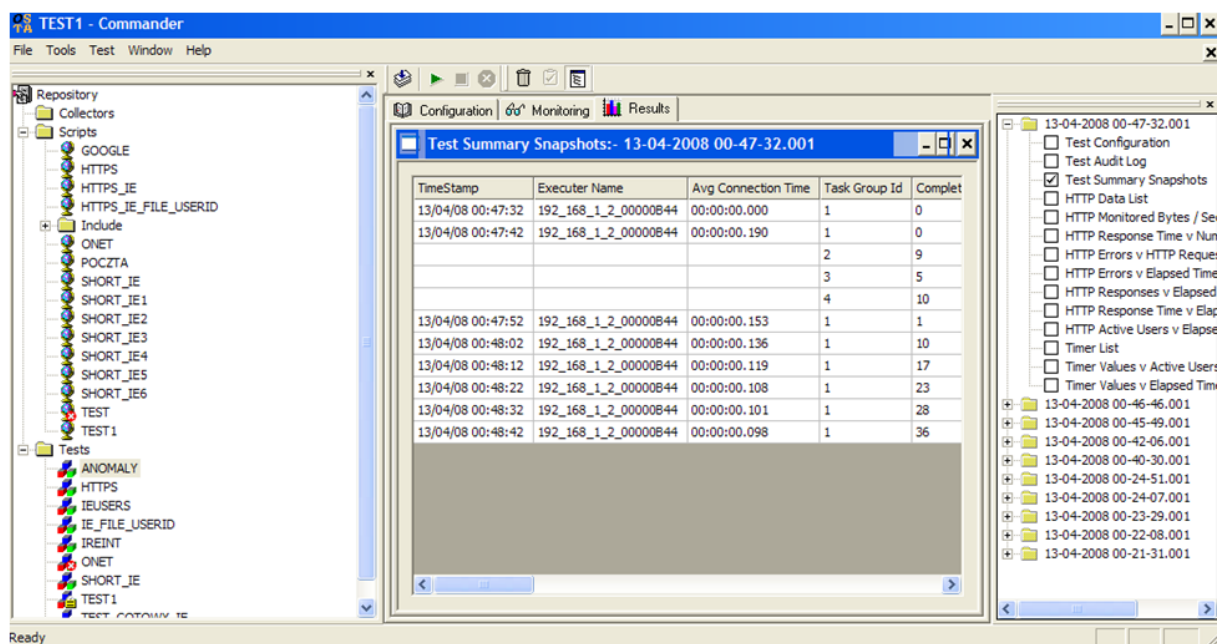
Rysunek 13 Ustawienie tasków

Generalnie, jeśli chodzi o użytkowników, to możemy też zaprogramować w pętli w kodzie np. zakładanie konta na stronie i obserwować, jak długo to trwa patrząc na różnego rodzaju raporty, które opiszę później. Teraz naciskamy na przycisk run i test zaczyna działać. Podczas wykonywania się testu możemy na bieżąco obserwować, co się tam dzieje. Służy do tego zakładka **Monitoring**. Po wejściu na nią, gdy test jest wykonywany możemy zaznaczyć np. zakładkę **Summary** i obserwować, jak test jest wykonywany. (Nie jest to jednak niestety przyjazne dla użytkownika, gdyż informacje nie do końca są czytelne). Ponadto możemy weryfikować, czy podczas wykonywania nie występuje żaden błąd.



Rysunek 14 Raporty podczas wykonywania się testu

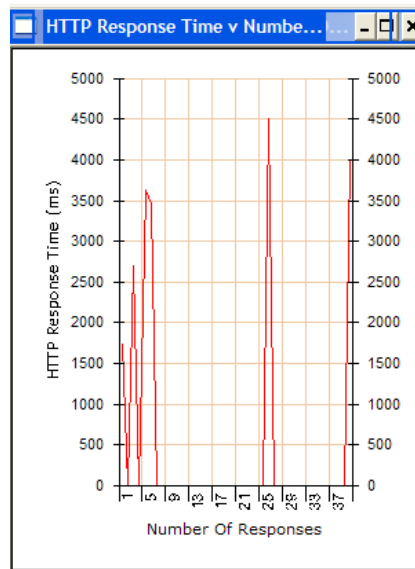
Po wykonaniu się testu nie pozostaje nam nic innego, jak zakładka **Results**. Znajduje się tam mnóstwo raportów, z których możemy dowiedzieć się między innymi o konfiguracji, czasach wykonywania iteracji, przeanalizować wykresy i przeglądać inne opcje.



Rysunek 15 Rezultaty po wykonaniu testu

Raporty wybieramy zaznaczając w drzewie po prawej stronie checkbox obok interesującej nas opcji. Myślę, że powinniśmy zapoznać się z każdą opcją w celu znalezienia potrzebnych nam informacji oraz zapoznania się z wykresami pokazującymi różne zależności. Jeśli podczas wykonywania scenariusza wystąpią błędy, zostanie automatycznie stworzony log z opisem błędów. Poniżej kilka screenów z raportów.

#	Time Stamp	User ID	URL	Response T...	Response C...	Reply Size
4	2008-04-13 00:47:32	2-9	GET http://www.g...	1156	200	7081
5	2008-04-13 00:47:32	2-10	GET http://www.g...	1203	200	7074
6	2008-04-13 00:47:32	2-4	GET http://www.g...	1265	200	7074
7	2008-04-13 00:47:32	2-3	GET http://www.g...	1312	200	6314
8	2008-04-13 00:47:32	2-11	GET http://www.g...	1312	200	7074
9	2008-04-13 00:47:32	3-3	GET http://www.g...	1343	200	7073
10	2008-04-13 00:47:32	2-8	GET http://www.g...	2031	200	7081
11	2008-04-13 00:47:32	3-1	GET http://www.g...	2156	200	7074
12	2008-04-13 00:47:32	2-12	GET http://www.g...	2250	200	7081
13	2008-04-13 00:47:32	2-7	GET http://www.g...	2312	200	6320
14	2008-04-13 00:47:32	2-1	GET http://www.g...	2484	200	6314
15	2008-04-13 00:47:32	2-5	GET http://www.g...	2593	200	7074
16	2008-04-13 00:47:32	4-1	GET http://www.g...	3890	200	7074
17	2008-04-13 00:47:32	4-4	GET http://www.g...	3875	200	7074
18	2008-04-13 00:47:32	4-11	GET http://www.g...	3953	200	7081
19	2008-04-13 00:47:32	4-19	GET http://www.g...	4046	200	6314
20	2008-04-13 00:47:32	3-2	GET http://www.g...	4140	200	7081
21	2008-04-13 00:47:32	4-6	GET http://www.g...	4375	200	7074
22	2008-04-13 00:47:32	4-10	GET http://www.g...	4406	200	7081
23	2008-04-13 00:47:32	4-5	GET http://www.g...	4734	200	7074
24	2008-04-13 00:47:32	4-7	GET http://www.g...	4828	200	6314
25	2008-04-13 00:47:32	4-9	GET http://www.g...	4859	200	7081
26	2008-04-13 00:47:32	4-17	GET http://www.g...	5031	200	7074
27	2008-04-13 00:47:32	4-18	GET http://www.g...	5078	200	7074
28	2008-04-13 00:47:32	3-7	GET http://www.g...	5312	200	6314
29	2008-04-13 00:47:32	3-5	GET http://www.g...	5359	200	7074



Rysunek 16 Przykładowe raporty

Opisywane powyżej możliwości OpenSta są tylko opisem tworzenia prostego scenariusza testowego. Mogą one być rozbudowywane poprzez dodawanie nowych funkcji w kodzie oraz poprzez zmianę ustawień konfiguracyjnych testów. Podczas wykonywania testów niestety nie widzimy, jak program „chodzi” po stronach, gdyż wszystkie operacje wykonywane są w tle.

Jak każdy program także OpenSta nie jest bez wad. Kilka z nich, które poznałem chciałbym opisać. Jedną z wad jest fakt, iż program nie zawsze chce nagrywać po HTTPS – w sytuacji wystąpienia takiego problemu pozostaje nam nagranie po http i potem zmiana w kodzie na HTTPS lub ponowne nagrywanie. Spowodowane jest to błędami podczas tworzenia programu i jak doczytałem na forum problem ten ma zniknąć w kolejnej wersji OpenSta. Kolejnym błędem jest fakt, że nieraz po uruchomieniu przeglądarki nie chce wyświetlać nam żadnej strony. Spowodowane jest to serwerem, który wpisuje sobie swoje Proxy na czas uruchamiania programu. Należy wtedy ponownie ustawić serwer Proxy, jak na załączonych powyżej obrazkach. Innym błędem, jaki wychyciłem był problem z długością znaków podczas nagrywania skryptu. Z tego powodu kompilacja nie była możliwa. Jako że mamy podane linie, w których są te błędy, pozostaje nam skrócenie tego stringa ( ja miałem problem z wersjami przeglądarek , skróciłem tylko do IE 7 i problem został wyeliminowany – String "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) ; .NET CLR 1.1.4322; InfoPath.2; MEGAUPLOAD 2.0; .NET CLR 2.0.50727)" ).

## Podsumowanie

Podsumowując uważam, że mimo pewnych wad jest to godny polecenia, szybko działający, posiadający dużo możliwości i ciągle rozwijany program do mierzenia wydajności. Programy takiego typu ułatwiają mi pracę i podczas testowania aplikacji zawsze zastanawiam się, jak można by wykorzystać jakiś program, aby zautomatyzować testy aplikacji, a gdy już mam to gotowe pozostaje mi myślenie nad optymalizacją. Chętnych do pogłębiania wiedzy na temat tego programu odsyłam do literatury:

<http://www.opensta.org/>

<http://portal.opensta.org/>