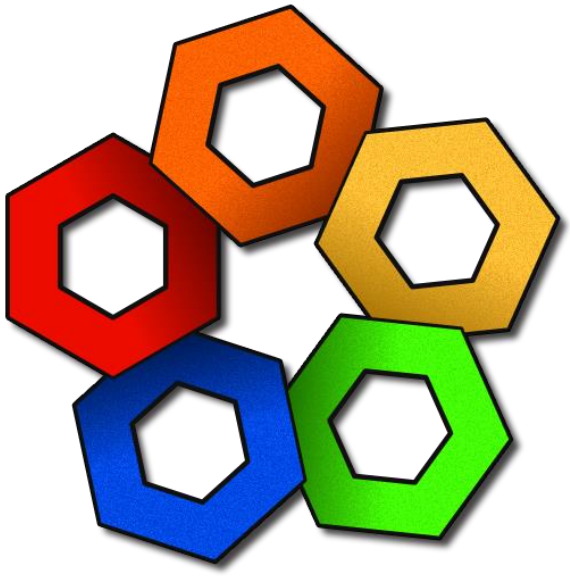

Page Analyzer Documentation

Author: Reto Scheiwiller

Installation

1. Extract the zip file.
2. Adjust the configuration under "./config/pageanalyzer.properties" to your needs.
3. Start the .jar-File by executing "start.bat" (you might have to add the java path in the file)

Setup Authentication

1. In the configuration file "./config/pageanalyzer.properties", change the property “pa_enable_authentication” to “true”.
2. Choose the source of the users by setting the value of “authentication_method”:
    a. **CSV:** Set user accounts in the file "./config/credentials.csv”.
    b. **LDAP:** Active Directory, define the other settings for LDAP in the properties file.
    c. **DB:** Internal DB only. Currently there is only the admin account available.

Access Page Analyzer

To access the application use the following URL:

- http://<yourservername>:<port>/pageanalyzer/app/login
  (for localhost http://localhost:8888/pageanalyzer/app/login)

Admin Account

The default admin account is as follows:

- **Username:** admin
- **Password:** admin

If you have changed and forgot the admin password of this account you can do the following to reset the password back to the default:

- In the "./config/cfw.properties", set the value of "cfw_reset_admin_pw" to true and restart the application. A log statement of level INFO will be written confirming the reset of the password. Change the setting to false after the restart to prevent subsequent resets of the password.
- Connect to the database and delete the admin account from the database. Restart the application, the admin account will be created again. This might delete as well all other data from the database related to the account.

API

The API can be accessed without authorization.

Analyze HAR

Key	Value
Description	Analyzes the HAR file provided in the request body. The will only be returned in the response and not saved on the server.
URL	http://localhost:8888/pageanalyzer/api/analyzehar
Type	POST
Parameters	• **harFile:** The content of the HAR file as multipart/form-data.
Returns	YSlow results in JSON format.

Custom Ruleset: pageanalyzer

Page analyzer uses its own ruleset, as there where several issues with the existing rules and yslow implementations in general:

Key	Value
cset.getComponentsByType() Issue	Components(~Requests) are selected by for the rule evaluation with the method getComponentsByType(). Most of the YSlow rules ignore components completely when they are requested after the onload method was executed. Therefore most of the rules did not a sufficient analysis, by adding the parameter "true" to the method this gets fixed.
Messages	Messages set by the rules where often not returned by the nodejs server. This was fixed by putting the messages in the "components" part of the results. Also some messages where missing important details.
Not checking if rule applicable	Most rules did not check if it is applicable to the analyzed page. For example if a page did not do any AJAX requests it still got 100% for this rule instead of "N/A". That increased the overall score of the page, what gave a wrong picture in the end.
Not working without pages	It was not working if no pages and only entries where defined in the .har-File (e.g. Fiddler export). This issue was fixed by iterating over all entries and ignore pages.
Only First Page Analyzed	Only the first page in the .har-File was analyzed with the out of the box yslow.js, it was changed to analyze all pages.
Ignoring Duplicated requests	addComponent() and addComponentNoDuplicates() both were throwing away duplicated requests, completely ignoring this part of the page.
pacacheajax	improved version of "yxhr".
paexpires	Improved version of "yexpires", the rule was not working properly as it took the current date and time when the yslow analysis was exectued, instead of using the time the page was actually loaded, what lead to false positives if you analyzed a .har-File which was created some time ago. The yexpires rule did also not consider when there was not expiration was set at all and did not blame in suhc cases. Also some details where added to the output(expiration timestamp and relative time).
paexternalcss **paexternaljavascript**	Rule "yexternal" was split up to get more distinction.

pafavicon	Improved version of "yfavicon", the rule was not working correctly as it extracted the favicon paths from the header(in most cases relative) and compared it to the components urls (often absolute), so it was not able to identify the icons.
pagetforajax	improved version of "yxhrmethod".
paimgnoscale	Improved version of "yimgnoscale".
pajsbottom	Improved version of "yjsbottom". the rule only checked javascripts in the <head>-Tag and not in hte <body>-Tag, therefore scripts might have not have been analyzed(also it did not check anything that came after on load). The yjsbottom rule did not consider inline scripts. Added some details to the results.
pamergejs **pamergecss**	Custom rules, check if it would make sense to merge javascript or css files.
pamincookie	Improved version of "ymincookie", the rule was only checking cookies of the main document and ignored all other cookies.
paminifycss **paminifyjavascript** **paminifyjson** **paminifyxml** **paminifyxhr**	Rule "yminify" was split up to get more distinction between css and javascript. The rules for other formats are custom rules. Also the YSlow method"isMinified" was improved to get better results.
panohttp4xx	improved version of "yno404". Will check for any HTTP 4xx status and will blame for it instead of only
paduplicatedrequests	Custom rule, checks if there are duplicated requests on the page.
paurllength	Custom rule, checks if there are requests with a URL longer than 255 bytes.
paetags	Replacement for “yetags”. The yetags rule only checked for specific ETag configuration of Apache and IIS servers. This rule checks for all resources if there are etags defined or not.