VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ANALÝZA A OZNÁMENÍ O NOVÝCH RESULTCLOUD VÝSLEDCÍCH

ANALYSIS AND NOTIFICATION OF NEW RESULTCLOUD SUBMISSIONS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

BOHDAN IAKYMETS

VEDOUCÍ PRÁCE
SUPERVISOR

Doc. RNDr. JMÉNO PŘÍJMENÍ, Ph.D.

BRNO 2016

Abstrakt

Abstract

Main goal of a project is design and make mechanism for analyzing and notify users about interesting changes in new uploaded submissions. Mechanism must support a few types of notifications and have possibility to extend types. Also changes must have interface for results presentation. Interested people would be able to set notifications about interesting analyze results. It is important and useful, cause biggest part of test results are not interesting and useless information, like same testing results. So main goal of analyzing is find the interesting results and show it to user.
Analyzing of submissions is very important, because a lot of results a useless, in most cases are the same data, so it don't any important information. Analyzing help save developer's time, it finds useful information and notify developer or other interested people about it, so anyone in any time can easyly find needed information, or see statistics of project.
But first of all, I must learn inner architecture of ResultCloud. How it works. It helps me better use all opportunities in design and programming that mechanism.

Klíčová slova

Keywords

Sem budou zapsána jednotlivá klíčová slova v anglickém jazyce, oddělená čárkami.

Citace

Bohdan Iakymets: Analýza a oznámení o nových ResultCloud výsledcích, bakalářská práce, Brno, FIT VUT v Brně, 2016

Analýza a oznámení o nových ResultCloud výsledcích

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana X... Další informace mi poskytli... Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

. . . .
Bohdan Iakymets
20. dubna 2016

Poděkování

V této sekci je možno uvést poděkování vedoucímu práce a těm, kteří poskytli odbornou pomoc (externí zadavatel, konzultant, apod.).

Obsah

Kapitola 1

Introduction in ResultCloud

1.1 What is ResultCloud

ResultCloud is a system for management of long-term testing results. It's means, ResultCloud collect testing results for some project and compare it, so developer can easy find difference between them. In now days doesn't exist new, modern instruments for collecting and showing in humans readable form testing results. How wrote Fillip Matys: "Tools which solve that problem [3], are too old and fall behind all modern applications. One of the biggest problem of that tools is no oppotrunity to expand and with growing market of different mobile devices with internet connection not able to present data in responsive form." So all instruments which we have now is too old, and doesn't extendable. But in ResultCloud prasing and management do modules. Each module is written for one type of testing results. For example module "DejaGnu summary v1.0" can parse and show only SystemTap results.

Data organization in ResultCloud is represent in Plugins. Plugins are modules, every plugin have own implementation of Parser, which parse input data, but save it in shared Project, Submission and etc entities. Hierarchy of shared entities is Projects contains Submissions, Submissions is a results of a single series of tests, which also divide to Categories, Categories are divided to TestCases, and TestCases to Results.

In this bachelor work I will analyzing Submissions, like their Results, and other stuff that may be interesting for people.

1.2 What is Submission

How I wrote on the top, submissions is a single testing result, the smallest part of every submission is Result, this part contain result of a single test from series of tests. All other part like Categories, TestCases are only organization unit.

There are two way to import new Submission, first is from web page, second is with the help of API. Then will include plugin for parse submission file. Every plugin has class Parser, for parsing files and put them to Database. Every plugin also has class Vizualization, for getting from DB submission data and return it in needed presentation. Vizualization can for graph, list or dashboard.

Kapitola 2

Mechanism design

Before start describing all possible kinds of analyzing, need think about how it possible easy add to whole system. So there must be some mechanism which would be like bridge between system and all kinds of analyzing.

That mechanism would get all existing kinds of analyzing and enable to use it. Mechanism also would provide entity for saving analyze data. Analyzer can't work with DB cause it's a good practice to divide work between separated modules, like "Divide and Conquer", so analyzer should only analyze input data and visualize it, that all. Centralized method good for that case because user don't need to load needed analyzer and work with DB, all this stuff do Analyze Controller.

Analyze controller in the beginning get all needed submissions (it can be all submissions of project or only some of them), plugin name and new uploaded submission. Then put all this parameters to analyzer classes, which after processing return results back to analyze controller, which is save it to database(DB). Analyzer can include a few methods of analyze for different plugins. If analyzer doesn't support current plugin it return empty result. Also in ResultCloud exist simple method of analyzing (compare new submission with previous one), after making analyzing controller it can be rewriting to individual analyze class.

Analyze controller would provide methods for setup available analyzers. Field "Results" in Analyze entity will contain text data which depend only on analyzer.

Analyzer also contain methods for processing saved results and return human readable result, in JSON. Analyze controller will collect all that results and it to client.

With ResultCloud it will connect in a few point, after uploading new submission plugin Parser will call Analyze Controller method for analyzing. And like in case with Dashboard, for analyze page it would have own service, which call visualization method and return data to client.

Analyzer controller call *analyze* only when new submission would be inserted into DB, in ImportService class.

Kapitola 3

Analyzer design

For right connecting with analyze controller, analyzer must have first of all static constant attribute with unique ID (under it ID, analyzer would be identified in Analyze entity), method for getting and processing data (name of the method must be the same in all Analyzer classes) and two functions for vizulizating data (Visualize, VisualizeSingle). Vizualizating functions will get data from AnalyzerController which get it from DB and return it in JSON.

Method for processing data get in parameters: array of submission, new submission and plugin name. It must return ValidationResult object, with string in Data attribute, or array of string if it has a few resuls, or it can return empty result, with null in Data attribute.

Kapitola 4

Analyzer vizualisation

Because different analyzers use different methods fo vizualization there is needed to specificate different possibilites to vizualize information. For that goal exist directory *templates* which contains folders for all analyzers with Angular derective and HTML templates. So component `analyzeOverview` always can upload all directives and for each analyzers results find correct view.

Kapitola 5

Kinds of Analyzer

Here's some kinds of analyzer which results would be interesting for programmers. - Find strange changes like if result has a long time the same value and than it change - Check a changes in tests, like if some test which is contained in all previous submissions just dissapear - Check if some test had a long sequence of some bad value like FAIL or ERROR and then take a PASS, but after take FAIL or ERROR again - Check changes from UNTESTED to some result - Check if presented a new tests - Check strange changes like FAIL -> ERROR - Check if count of bad results is get maximum

Kapitola 6

Notification design

Like in case with Analyze Controller, I would divide notifications methods to the separated classes and Notification Controller will controller them. NotificationController will collect all Notifiers with the help of static function *preLoad*. For notifing exist function *notify(title, body, bodyShort, to)*, where `title` - is title of notification message, `body` - longest body text, `bodyShort` - short body of the message not longer than 140 letters, `to` - is an array of all adresses with the key of notifier ID. Function *notify* calling notifier's function *notify*, only for notifiers which have their IDs in `to` parameter's key.

Also notifications divided to: Public and Private. Public notifications, is notifications which would be sent to public resources, like RSS or Twitter. Private notifications would be sent to private persons, like e-mail for exammple.

Private notifications have settings for each user. And user can subscribe to each private notification.

For Twitter notification is using TwitterAPIExchange with MIT license.

Kapitola 7

Notifier

Notifier is class for notification users in specific way. Notifier class is extend from BaseNotifier class.

BaseNotifier class has only one function which will be in most cases the same in private notification. This is *getSettings* function, which will returning needed settings field for each private notifier.

Notifier also have function *notify*, which get parameters from NotificationController and send all needed notification messages.

Kapitola 8

Notifiers

In that moment in ResultCloud are Email, Twitter and RSS notifiers.

Přílohy

Seznam příloh