|  |
| --- |
| // ==UserScript== |
|  | // @name Quizlet Match Hack |
|  | // @namespace Thanos de Slayer |
|  | // @version 1.6 |
|  | // @description The time will freeze at 0.5 and the answers will be the same color |
|  | // @author You |
|  | // @match https://quizlet.com/\*/\* |
|  | // @grant none |
|  | // @license MIT |
|  | // ==/UserScript== |
|  |  |
|  | //Ignore the variable gravityScore because it doesn't work any more, but I decided to keep the code here |
|  | var gravityScore, |
|  | href = window.location.href; |
|  | //<br><br><button class="UIButton" id="customWaitButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button> |
|  | (function() { |
|  | initLoad(); |
|  | function initLoad() { |
|  | if (href.includes("quizlet.com")) { |
|  | if (href.includes("/learn")) { |
|  | cAlert('<h2>Game Mode: Learn</h2>Thank you for using gabes Quizlet Exploit<br>Dont bother changing the time it only goes to 0.5<br><h4>Instructions:</h4>Just wait for this script to finish!<br><br><button class="UIButton" id="learnButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button>'); |
|  | getId("learnButton").addEventListener("click", function() { |
|  | document.getElementById("customMessageContainer").remove(); |
|  | learn(); |
|  | }); |
|  |  |
|  | } else if (href.includes("/flashcards")) { |
|  | cAlert('<h2>Game Mode: Flashcards</h2>Thank you for using Thanos Quizlet Exploit<br>Dont bother changing the time it only goes to 0.5<br><h4>Changelog:</h4>+ Added Match time freeze for regular match and diagrams<br>+ Added Gravity score exploit to get ANY score you want!<br>+ Added custom alert box<br>+ Fixed graphics<br>- Removed useless alert boxes.<h4>Instructions:</h4>Umm why are you here? Go cheat somewhere else...<br><br><button class="UIButton" id="flashcardsButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button>'); |
|  | getId("flashcardsButton").addEventListener("click", function() { |
|  | document.getElementById("customMessageContainer").remove(); |
|  | }); |
|  | } else if (href.includes("/write")) { |
|  | cAlert('<h2>Game Mode: Write</h2>Thank you for using Thanos Exploit<br>Dont bother changing the time it only goes to 0.5<br><h4>Instructions:</h4>You dont even have to wait,<br> this is my favorite one to watch!<br><br><button class="UIButton" id="writeButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button>'); |
|  | getId("writeButton").addEventListener("click", function() { |
|  | document.getElementById("customMessageContainer").remove(); |
|  | write(); |
|  | }); |
|  | } else if (href.includes("/spell")) { |
|  | cAlert('<h2>Game Mode: Spell</h2>Thank you for using Thanos Quizlet Exploit<br>Dont bother changing the time it only goes to 0.5.<br><h4>Instructions:</h4>Hold down enter and wait.<br><br><button class="UIButton" id="spellButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button>'); |
|  | getId("spellButton").addEventListener("click", function() { |
|  | document.getElementById("customMessageContainer").remove(); |
|  | spell(); |
|  | }); |
|  | } else if (href.includes("/test")) { |
|  | cAlert('<h2>Game Mode: Test</h2>Thank you for using Thanos Quizlet Exploit<br>Dont bother changing the time it only goes to 0.5<br><br><h4>Instructions:</h4>Right click to toggle answers.<br>(Be subtle when using)<br><br><button class="UIButton" id="testButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button>'); |
|  | getId("testButton").addEventListener("click", function() { |
|  | document.getElementById("customMessageContainer").remove(); |
|  | testMode(); |
|  | }); |
|  | } else if (href.includes("/micromatch")) { |
|  | cAlert('<h2>Game Mode: Micromatch</h2>Thank you for using Gabes Quizlet Exploit<br>Dont bother changing the time it only goes to 0.5<br><h4>Instructions:</h4>The timer will be paused when at choosen time.<br>The answers will also be highlighted for you.<br>At your leisure, solve the questions.<br><h4>Match Time: </h4><input type="text" id="matchTimeInput" value="0"></input><br><br><button class="UIButton" id="micromatchButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button>'); |
|  | getId("micromatchButton").addEventListener("click", function() { |
|  | document.getElementById("customMessageContainer").remove(); |
|  | micromatch(); |
|  | }); |
|  | } else if (href.includes("/match")) { |
|  | cAlert('<h2>Game Mode: Match</h2>Thank you for using Gabes Quizlet Exploit<br>Dont bother changing the time it only goes to 0.5<br><h4>Instructions:</h4>The timer will be paused when at choosen time.<br>The answers will also be highlighted for you.<br>At your leisure, solve the questions.<br><h4>Match Time: </h4><input type="text" id="matchTimeInput" value="0"></input><br><br><button class="UIButton" id="matchButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button>'); |
|  | getId("matchButton").addEventListener("click", function() { |
|  | document.getElementById("customMessageContainer").remove(); |
|  | match(); |
|  | }); |
|  | } else if (href.includes("/gravity")) { |
|  | try { |
|  | document.getElementsByClassName("UIModal is-white is-open")[0].style.display = "none" |
|  | } catch (e) {} |
|  | cAlert('<h2>Game Mode: Gravity</h2>Thank you for using Gabes Quizlet Exploit<br>Dont bother changing the time it only goes to 0.5.<br> <h4>What score would you like?: <input type="number" id="gravityScoreInput" value="4294967295" onchange="gravityScore = parseInt(this.value);"></input></h4><br><br><button class="UIButton" id="gravityButton" type="button"><span class="UIButton-wrapper"><span>Inject</span></span></button>'); |
|  | getId("gravityButton").addEventListener("click", function() { |
|  | document.getElementById("customMessageContainer").remove(); |
|  | gravity(); |
|  | }); |
|  |  |
|  | } else { |
|  | } |
|  | } |
|  |  |
|  | function testMode() { |
|  | var question = getClass("TermText notranslate lang-en"); |
|  | for (var i = 0; i < question.length; i++) { |
|  | question[i].innerHTML += '<br><small style="font-weight: bold; display: block;" class="answers">' + findAnswerGlobal(question[i].innerHTML) + "</small>"; |
|  | } |
|  | window.oncontextmenu = function(e) { |
|  | e.preventDefault(); |
|  | var answer = getClass("answers"); |
|  | if (answer[0].style.display == "block") { |
|  | for (var i = 0; i < answer.length; i++) { |
|  | answer[i].style.display = "none"; |
|  | } |
|  | } else { |
|  | for (var i = 0; i < answer.length; i++) { |
|  | answer[i].style.display = "block" |
|  | } |
|  | } |
|  | } |
|  | } |
|  |  |
|  | function gravity() { |
|  | if (gravityScore === undefined) { |
|  | gravityScore = parseInt(prompt("Enter exact score:")); |
|  | } |
|  |  |
|  | var t = window.QJP([], [], ["hyek"]).a, |
|  | n = t.grader.grade.bind(t.grader); |
|  | t.\_startGame(), t.dataMap = t.dataMap.update("points", function(t) { |
|  | return gravityScore; |
|  | }); |
|  | for (var o = 0; o < 99; o++) t.\_advanceLevel(); |
|  |  |
|  | function s(e) { |
|  | t.grader.grade = function(e, t, n) { |
|  | return !0; |
|  | }, e(), t.grader.grade = n |
|  | } |
|  | s(function() { |
|  | console.log(null); |
|  | }); |
|  | var a = t.\_promptCopyAnswer.bind(t); |
|  | t.\_promptCopyAnswer = function(e) { |
|  | a(e), setTimeout(function() { |
|  | s(function() { |
|  | t.\_checkCopiedAnswer({ |
|  | liveTermId: e, |
|  | answer: "" |
|  | }); |
|  | }); |
|  | }, 0); |
|  | }; |
|  | } |
|  |  |
|  | function write() { |
|  | //This is sad... but works better... |
|  | var remaining = parseInt(document.getElementsByClassName("LearnModeProgressBar-value")[0].innerHTML); |
|  | for (var i = 0; i < remaining; i++) { |
|  | getId("user-answer").value = Math.random(); |
|  | getId("js-learnModeAnswerButton").click(); |
|  | getClass("js-learnModeOverrideIncorrect")[0].click(); |
|  | } |
|  | /\* |
|  | var buttons = document.querySelectorAll("button"), |
|  | span = document.querySelectorAll("span"); |
|  |  |
|  | if (getId("user-answer")) { |
|  | getId("user-answer").disabled = true; |
|  | getId("user-answer").value = findAnswerGlobal(getClass("qDef lang-en TermText")[0].innerHTML); |
|  | for (var i = 0; i < buttons.length; i++) { |
|  | if (buttons[i].childNodes[0].innerHTML == "Answer") { |
|  | buttons[i].click(); |
|  | } |
|  | } |
|  | try { |
|  | for (var i = 0; i < span.length; i++) { |
|  | if (span[i].childNodes[0].childNodes[0].innerHTML == "Override: I was right") { |
|  | span[i].click; |
|  | } |
|  | } |
|  | } catch (e) {} |
|  | write(); |
|  | } else { |
|  | for (var i = 0; i < buttons.length; i++) { |
|  | if (buttons[i].innerHTML == "Press any key to continue") { |
|  | buttons[i].click(); |
|  | } else if (buttons[i].innerHTML == "Start Over") { |
|  | return 1; |
|  | } |
|  | } |
|  | setTimeout(write, 0); |
|  | } |
|  | \*/ |
|  | } |
|  |  |
|  | function spell() { |
|  | if (getClass("SpellModeControls-progressValue")[0].innerHTML == "100%") { |
|  | return 1; |
|  | } |
|  | if (getId("js-spellInput") !== null) { |
|  | getId("js-spellInput").value = findAnswerGlobal(getClass("qDef lang-en TermText")[0].innerHTML); |
|  | setTimeout(spell, 10); |
|  | } |
|  | } |
|  |  |
|  | function match() { |
|  | button = getClass("UIButton UIButton--hero")[0], button && button.click(); |
|  | setTimeout(function() { |
|  | var matchLoop = setInterval(function() { |
|  | waitForMatch(); |
|  | }, 0); |
|  |  |
|  | function waitForMatch() { |
|  | if (getClass("MatchModeQuestionScatterTile") || getClass("MatchModeQuestionGridBoard-tile")) { |
|  | for (var F = setTimeout(";"), i = 0; i < F; i++) clearTimeout(i); |
|  | var tiles = getClass("MatchModeQuestionScatterTile"); |
|  | var colors = ["#FF0000", "#FF0000", "#FF6600", "#FF6600", "#FFFF00", "#FFFF00", "#00FF00", "#00FF00", "#00FFFF", "#00FFFF", "#0033FF", "#0033FF", "#FF00FF", "#FF00FF", "#CC00FF", "#CC00FF", "#6E0DD0", "#6E0DD0", "#C0C0C0", "#C0C0C0", "#FFFFFF", "#FFFFFF", "#A52A2A", "#A52A2A", "#F6CFFF", "#F6CFFF", "#CFD9FF", "#CFD9FF", "#FBFFA3", "#FBFFA3", "#FFD1A3", "#FFD1A3", "#710000", "#710000"]; |
|  | for (var i = 0; i < tiles.length; i++) { |
|  | tiles[i].style.backgroundColor = colors[i]; |
|  | for (var j = 0; j < tiles.length; j++) { |
|  | if (tiles[j].childNodes[0].innerHTML == findAnswerGlobal(tiles[j].childNodes[0].innerHTML)) { |
|  | tiles[j].style.backgroundColor = colors[i]; |
|  | } |
|  | } |
|  | } |
|  | clearTimeout(matchLoop); |
|  | } |
|  | } |
|  | }, 500); |
|  | } |
|  |  |
|  | function micromatch() { |
|  | button = getClass("UIButton UIButton--hero")[0], button && button.click(); |
|  | setTimeout(function() { |
|  | setTimeout(function() { |
|  | for (var F = setTimeout(";"), o = 0; o < F; o++) clearTimeout(o); |
|  | }, 0); |
|  |  |
|  | var tiles = getClass("MatchModeQuestionGridTile-text"); //[0].childNodes[0].innerHTML |
|  | for (var i = 0; i < tiles.length; i++) { |
|  |  |
|  | if (getClass("MatchModeQuestionGridTile")[i].classList[1] != "is-selected") { |
|  | click(getClass("MatchModeQuestionGridBoard-tile")[i].childNodes[0], "pointerdown"); |
|  | } |
|  | for (var j = 0; j < tiles.length; j++) { |
|  | if (tiles[j].childNodes[0].innerHTML == findAnswerGlobal(tiles[i].childNodes[0].innerHTML)) { |
|  | if (getClass("MatchModeQuestionGridTile")[j].classList[1] != "is-selected") { |
|  | click(getClass("MatchModeQuestionGridBoard-tile")[j].childNodes[0], "pointerdown"); |
|  | j = tiles.length; |
|  | } |
|  | } |
|  | } |
|  | } |
|  | function click(e, t) { |
|  | if (e.fireEvent) e.fireEvent("on" + t); |
|  | else { |
|  | var n = document.createEvent("Events"); |
|  | n.initEvent(t, !0, !1), e.dispatchEvent(n); |
|  | } |
|  | } |
|  | }, 500); |
|  | } |
|  |  |
|  | function learn() { |
|  | if (getClass("ProgressSegmentedSemicircle-percent")[0].innerHTML = "100%") { |
|  | return 1; |
|  | } |
|  | checkCheckbox(); |
|  | if (getClass("AssistantMultipleChoiceQuestionPromptView-inner")[0] !== undefined) { |
|  | var answer = findAnswerGlobal(getClass("FormattedText notranslate TermText lang-en")[0].innerHTML); |
|  | var q1 = getClass("FormattedText notranslate TermText lang-en")[1]; |
|  | var q2 = getClass("FormattedText notranslate TermText lang-en")[2]; |
|  | var q3 = getClass("FormattedText notranslate TermText lang-en")[3]; |
|  | var q4 = getClass("FormattedText notranslate TermText lang-en")[4]; |
|  | if (answer == q1.innerHTML) { |
|  | q1.click(); |
|  | } else if (answer == q2.innerHTML) { |
|  | q2.click(); |
|  | } else if (answer == q3.innerHTML) { |
|  | q3.click(); |
|  | } else if (answer == q4.innerHTML) { |
|  | q4.click(); |
|  | } else { |
|  | console.error("ERROR: Unable to find / click answer"); |
|  | return 0; |
|  | } |
|  | getClass("UIButton")[4].click(); |
|  | setTimeout(learn, 10); |
|  | } else { |
|  | setTimeout(learn, 100); |
|  | } |
|  | } |
|  |  |
|  | function checkCheckbox() { |
|  | document.getElementsByClassName("UIButton UIButton--whiteBorder UIButton--fill")[0].click(); |
|  | if (document.getElementsByClassName("UICheckbox-input")[3].checked && document.getElementsByClassName("UICheckbox-input")[2].checked !== true && document.getElementsByClassName("UICheckbox-input")[4].checked !== true) { |
|  | document.getElementsByClassName("UIButton UIButton--inverted")[0].click(); |
|  | return 1; |
|  | } |
|  | if (document.getElementsByClassName("UICheckbox-input")[4].checked === true) { |
|  | document.getElementsByClassName("UICheckbox-input")[4].click(); |
|  | } |
|  | if (document.getElementsByClassName("UICheckbox-input")[2].checked === true) { |
|  | document.getElementsByClassName("UICheckbox-input")[2].click(); |
|  | } |
|  | if (document.getElementsByClassName("UICheckbox-input")[3].checked === false) { |
|  | document.getElementsByClassName("UICheckbox-input")[3].click(); |
|  | } |
|  | document.getElementsByClassName("UIButton UIButton--inverted")[0].click(); |
|  | } |
|  |  |
|  | function findAnswerGlobal(question) { |
|  | if (Quizlet.assistantModeData !== undefined) { //Quizlet.assistantModeData.terms |
|  | return getAnswer(Quizlet.assistantModeData.terms, question); |
|  | } else if (Quizlet.learnGameData !== undefined) { //Quizlet.learnGameData.allTerms |
|  | return getAnswer(Quizlet.learnGameData.allTerms, question); |
|  | } else if (Quizlet.testModeData !== undefined) { //Quizlet.testModeData.terms |
|  | return getAnswer(Quizlet.testModeData.terms, question); |
|  | } else if (Quizlet.spellModeData !== undefined) { //Quizlet.spellModeData.spellGameData.termsById |
|  | return getAnswer(Quizlet.spellModeData.spellGameData.termsById, question); |
|  | } else if (Quizlet.matchModeData !== undefined) { //Quizlet.matchModeData.terms |
|  | return getAnswer(Quizlet.matchModeData.terms, question); |
|  | } else if (Quizlet.gravityModeData !== undefined) { //Quizlet.gravityModeData.terms |
|  | return getAnswer(Quizlet.gravityModeData.terms, question); |
|  | } else { |
|  | return 0; |
|  | } |
|  |  |
|  | function getAnswer(s, t) { |
|  | var e = s; |
|  | if (t.indexOf("\_") != "-1") { |
|  | if (t.slice(-1) == "\_") { //Underscore at end |
|  | for (var i=0; i<e.length; i++) { |
|  | if (e[i].definition.indexOf(getClass("qDef lang-en TermText")[0].innerHTML.split("\_")[0]) != "-1") { |
|  | return e[i].word; |
|  | } else if (e[i].word.indexOf(getClass("qDef lang-en TermText")[0].innerHTML.split("\_")[0]) != "-1") { |
|  | return e[i].definition; |
|  | } |
|  | } |
|  | } else if (t[0] == "\_") { |
|  | for (var i=0; i<e.length; i++) { //Underscore at start |
|  | if (e[i].definition.indexOf(getClass("qDef lang-en TermText")[0].innerHTML.split("\_").slice(-1)[0]) != "-1") { |
|  | return e[i].word; |
|  | } else if (e[i].word.indexOf(getClass("qDef lang-en TermText")[0].innerHTML.split("\_").slice(-1)[0]) != "-1") { |
|  | return e[i].definition; |
|  | } |
|  | } |
|  | } else { |
|  | for (var i=0; i<e.length; i++) { //Underscore in middle |
|  | if (e[i].definition.indexOf(getClass("qDef lang-en TermText")[0].innerHTML.split("\_").slice(-1)[0]) != "-1" && e[i].definition.indexOf(getClass("qDef lang-en TermText")[0].innerHTML.split("\_")[0]) != "-1") { |
|  | return e[i].word; |
|  | } else if (e[i].word.indexOf(getClass("qDef lang-en TermText")[0].innerHTML.split("\_").slice(-1)[0]) != "-1" && e[i].word.indexOf(getClass("qDef lang-en TermText")[0].innerHTML.split("\_")[0]) != "-1") { |
|  | return e[i].definition; |
|  | } |
|  | } |
|  | } |
|  | } |
|  | var answers = []; |
|  | for (var i=0; i<e.length; i++) { |
|  | e[i].definition = e[i].definition.replace("\n", "<br>"); |
|  | e[i].word = e[i].word.replace("\n", "<br>"); |
|  | if (t == e[i].word) { |
|  | answers.push(e[i].definition); |
|  | } else if (t == e[i].definition) { |
|  | answers.push(e[i].word); |
|  | } |
|  | } |
|  | return answers[Math.floor(Math.random() \* answers.length)]; |
|  | } |
|  | } |
|  | } |
|  |  |
|  | function cAlert(message) { |
|  | var html = '<div class="UIModal is-white is-open" id="customMessageContainer" role="document" tabindex="-1"> <div class="UIModal-box"> <div class="UIModalHeader"> <div class="UIModalHeader-wrapper"> <span class="UIModalHeader-close"> <div class="UIModalHeader-closeIconButton"> <span class="UIIconButton"> <button class="UIButton UIButton--inverted" type="button" id="customCloseButton" onclick="document.getElementById(&quot;customMessageContainer&quot;).remove();"> <span class="UIButton-wrapper"> <svg class="UIIcon UIIcon--x-thin"> <noscript></noscript> <use xlink:href="#x-thin"></use> <noscript></noscript> </svg> </span> </button> </span> </div> </span> <div class="UIModalHeader-childrenWrapper"> <h3 class="UIHeading UIHeading--three"><span id="customTitle">Gabes Quizlet Exploit</span></h3> </div> </div> </div> <div class="UIModalBody"> <div class="UIDiv SetPageEmbedModal-content"> <div> <p class="UIParagraph"><span id="customMessage">'+message+'</span></p></div></div></div></div></div>'; |
|  | var j = document.createElement('div'); |
|  | j.innerHTML = html; |
|  | document.body.appendChild(j); |
|  | } |
|  | })(); |
|  |  |
|  | function getId(id) { |
|  | return document.getElementById(id); |
|  | } |
|  |  |
|  | function getClass(id) { |
|  | return document.getElementsByClassName(id); |
|  | } |
|  |  |
|  | //var learnInsertJs = "function figure(match) {termsWord = window.Quizlet.assistantModeData.terms.filter(function z(x) {return x.word == match}); termsDefinition = window.Quizlet.assistantModeData.terms.filter(function z(x) {return x.definition === match}); if (termsWord.length > 0) {return termsWord[0].definition;}; if (termsDefinition.length > 0) {return termsDefinition[0].word;}}; function solve() {input = document.querySelector('#AssistantModeTarget > div > div > div > div.ModeLayout-content > div > span:nth-child(2) > div > div > div.AssistantScrollableViewLayout-content > div > div.AssistantMultipleChoiceQuestionPromptView-promptArea > div > div > div > span').innerHTML; output = figure(input); outs = document.querySelectorAll('.AssistantMultipleChoiceQuestionPromptView-termOptionInner > div > span'); for (var i = 0; i < outs.length; i++) {if (outs[i].innerText === output) {outs[i].click();}}; document.querySelector('#AssistantModeTarget > div > div > div > div.ModeLayout-content > div > span:nth-child(3) > div > div.AssistantFixedActionLayout-action > div > button').click(); if (document.querySelector('#AssistantModeTarget > div > div > div > div.ModeLayout-controls > div > div > div > div.ModeControls-main > div.ModeControls-progress > div > div > div > svg > text.ProgressSegmentedSemicircle-text.ProgressSegmentedSemicircle-progressText > tspan').innerHTML.slice(0, -1) !== '100') {setTimeout(solve, 10);};}; solve();" |
|  |  |
|  | //backupOptionsSet = "/\*document.querySelector('#AssistantModeTarget > div > div > div > div.ModeLayout-controls > div > div > div > div.ModeControls-main > div.ModeControls-actions > div > button > span > span').click(); if (document.querySelector('body > div:nth-child(4) > div.UIModal.is-white.is-open > div > div.UIModalBody > div > div:nth-child(3) > div > div > div > div > div > div:nth-child(3) > label > input').checked) {document.querySelector('body > div:nth-child(4) > div.UIModal.is-white.is-open > div > div.UIModalBody > div > div:nth-child(3) > div > div > div > div > div > div:nth-child(3) > label > input').click();}; if (document.querySelector('body > div:nth-child(4) > div.UIModal.is-white.is-open > div > div.UIModalBody > div > div:nth-child(3) > div > div > div > div > div > div:nth-child(1) > label > input').checked) {document.querySelector('body > div:nth-child(4) > div.UIModal.is-white.is-open > div > div.UIModalBody > div > div:nth-child(3) > div > div > div > div > div > div:nth-child(1) > label > input').click();}; document.querySelector('body > div:nth-child(4) > div.UIModal.is-white.is-open > div > div.UIModalHeader > div > span > div > span > button > span').click();\*/" |
|  | /\* |
|  | function solveLearn() { |
|  | script = document.createElement('script'); |
|  | script.textContent = learnInsertJs; |
|  | (document.head||document.documentElement).appendChild(script); |
|  | script.remove(); |
|  | } |
|  |  |
|  | chrome.runtime.onMessage.addListener( |
|  | function(request, sender, sendResponse) { |
|  | if (request.run == "write") { |
|  | iters = parseInt($(".LearnModeProgressBar-value").html()) |
|  | for (var i = 0; i < iters; i++) { |
|  |  |
|  | $("#user-answer")[0].value = "\*"; |
|  | $("#js-learnModeAnswerButton")[0].click(); |
|  | document.evaluate('//\*[@id="js-learnModeInner"]/div[1]/div/div[2]/div/div[2]/span/a', document, null, XPathResult.FIRST\_ORDERED\_NODE\_TYPE, null).singleNodeValue.click(); |
|  | } |
|  | sendResponse({out: 0}); |
|  | } |
|  | else if (request.run === "learn") { |
|  | solveLearn(); |
|  | sendResponse({out: 0}); |
|  | } |
|  | else { |
|  | sendResponse({out: 1}); |
|  | } |
|  | }); |
|  | \*/ |