Documentation

Install:

A way to install Watcheezy, is to copy and paste the line of code - provided by email - on the HTML source code of your website:

<script type="text/javascript" src="**********" async></script>

This line of code has to be placed on each page of the site where the service has to display watcheezy.

This line of code is made of javascript language: scripts are loaded in the order they are writen in the source code of the page. So the Watcheezy script may be placed at the bottom of the page source-code, after the other scripts and before the closing body tag </body>.

Example of html code :

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="X-UA-Compatible" content="chrome=1">
<title>Yout page title</title>
</head>
<body>

<p> Lorem ipsum dolor sit amet, consectetuer ...</p>

<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=******" async></script>
</body>
</html>
```

If the page already contains javascript errors, or if the behaviour of the website is altered, it is better to place Watcheezy's line of code at the beginning of the source code, above the header (between <head> tags). Then, the Watcheezy script must be the first script called on the page. In this case, a parameter "install" on the line must be set to "header".

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="X-UA-Compatible" content="chrome=1">
<title>Yout page title</title>
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=******&install=header" async></script>
<script type="text/javascript" src=myotherscript.js"></script>
</head>
<body>
<p> Lorem ipsum dolor sit amet, consectetuer ...</p>
</body>
</html>
```

If your are using a CMS to manage your website, the process is quite the same. In your CMS backend you have to map the Watcheezy line of code on all page you need. Depending on the CMS, you can choose where to install the scirpt in the page source code. As recommanded earlier, it is better to put Watcheezy script at the end of the page, as the last script. For example in Wordpress (with admin rights), go to “Editor” in “Appearance” menu, then copy the watcheezy line of code just before “/body”.

There are existing modules that automatically install Watcheezy on the CMS page (prestashop, wordpress...). In that case, all you have to do is to get your licence key and put it in the right field (only one time) in the CMS administration backend.

Warning :

As Watcheezy is made of javascript, the visitor's browser has to accept javascript. This could be set in the browser options.

As watcheezy is using third party network, the client network has to allow several ports to ensure the chat communication. The following URL/ports have to be opened (in the client firewall/proxy settings):

www.watcheezy.net => UDP 443, TCP 443, TCP 80
www.watcheebox.net => UDP 443, TCP 443, TCP 80 + 3000

The use of the service is optimal for the latest versions of Chrome and Firefox.

As Watcheezy use cookies on the visitor's system, they must be enabled. It is your business to inform the visitor about the use of cookie on your website. Depending on the country you live, it may be a legal obligation.

Parameters of the line of code (pseudo-api):

The Watcheezy backoffice allows to set some options to the service (tab color, opening options...) whatever the page is loaded. Thanks to this pseudo-api, it is possible to overwrite those options and/or apply special feature to target pages. To do this, there are several parameters that could be queued in the watcheezy script line of code. Those parameters apply some functionalities on the page they are set (and only this page).

Some of those parameter are required, and some else are optionnal.

Required parameters:

```
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=******..."></script>
```

key: it corresponds to the license key of the site. Please note that this value is unique and valid only for one site and its sub domain "www" if it exists. For example, the 2 main domains that share the same licence key are http://mywebsite.com and http://www.mywebsite.com,

Optional parameters:

```
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=****&install=header..."></script>
```

install: it defines whether if the scirpt is placed in the early source-code of the page. If the script is placed at the beginning of the source code (first script in <head>), the value must be set to "header".

```
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=***&install=***&lang=EN..."></script>
```

lang: defines the default langage of watcheezy (front and backoffice). The value (case sensitive) could be "FR" for french, "EN" for English, “DE” for german, “ES” for Spanish and “IT” for italian. If this parameter is not set, the default language will be the browser's one (or default english)

```
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=***&install=***&timeout=60..."></script>
```

timeout: defines, in seconds, a delay before loading the service. During this period watcheezy is not loaded at all, then it does not track the visitor or display the communication tab. The value must be between 0 and 300. At the end of the countdown, if the visitor is still online, the watcheezy tab appears automatically This option is used if you don't want the service to appears immediately. Warning: for this page, this parameter overwrite any "opening options" eventually set in your backoffice.

```
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=***&install=***&protocol=SSL..."></script>
```

protocol: defines whether the service is used in SSL mode. If the website use a secure protocole, the Watcheezy service will only appear if this parameter is set to "SSL". You can use this parameter even if your website is not a https one. However if your website is https, you are not forced to use this parameter,

```
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=***&install=***&hide=interpel..."></script>
```

hide: defines an opening option on the watcheezy tab. Thank to this parameter, you can choose not to display the tab on the page or display it only if an agent interpellate the visitor. If the parameter is set to "total" the visitor is tracked but not interpellable (and he cannot chat with an agent): the tab is not loaded on the page but his historic is saved. If the parameter is set to "interpel", the tab is loaded but invisible until an agent write to the visitor (thanks to the chat). Any other value do not change the service opening.

```
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=***&install=***&autopen=on..."></script>
```

autopen: defines if the tab is automatically open on the page: ie there is no need to click on the tab title to display the complete service. If the parameter is set to "on", watcheezy is fully opened, by default, on the page. The visitor can still hide the tab, at any time, clicking on the tab title.

How to :

With a little bit of javascript, you can easily set those values dynamically. For example, you want the watcheezy tab to appear only if your visitor is authorized by your own process (logged user), you can then script this :

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="X-UA-Compatible" content="chrome=1">
<title>welcome to logged user</title>
<script type="text/javascript">

var userRegistered = 0;
…//any code that tests if user is logged
userRegistered = 1;

var hideWatcheezy = "";
if ( userRegistered = = 0)
    hideWatcheezy = "hide";

…//rest of code

</script>
<script type="text/javascript" src=otherscripts.js"></script>
</head>
<body>
<p> Lorem ipsum dolor sit amet, consectetuer ...</p>

<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=******&install=footer&hide="+hideWatcheezy></script>

</body>
</html>
```

this code will make watcheezy tab to appear only if the parameter userRegistered is set to 1 by your own script...

API:

This API allows you to had special feature to basic Watcheezy service. There are three kind of customization then allowed:

- Adding custom criteria for the visitors targeting;
- Adding information about your visitor for better monitoring . Those information belong to you (your database, external API...);
- Adding condition for the watcheezy tab opening/appearance.

The scope of these functions is limited for the page where they are called and for the visitor that has loaded the page.

The use of API functions overpass any parameters used in the pseudo-api of the watchezy.js line of code, especially for concurrent functionalities...

Your part of the job is to get this information from your database, third party API or visitor session, then push those information in the Watcheezy API functions. This documentation does not assume the technology used to inject such data in the API (few PHP examples are provided).

For the time being, the API is mainly used in a synchrone mode. This means that the API functions, unless otherwise specified, are set once at the page loading. Functions are available only after the load of th main watcheezy script,

Using the API in 3 steps:

You fisrt need to prepare your criteria, data and/or opening conditions. This step require to define label, value and coding parameters: for example, you want an alert to be triggered in the Watcheezy backoffice each time a visitor log-in your website. You have to prepare the following label “visitor has logged”, and a server-side data or a session parameter that confirms that the user is logged at the loading of the page. Those data must be converted in javascript langage, and placed at the beginning of the page (before any code of Watcheezy API),

1 Preparing your criteria:

It is not needed to rewrite all your custom criteria on each page, you can prepare your script and put it in a js file, and then include it before / after watcheezy.js

myscript.js

```
<!--
forceWatcheeAlert("interesting visitor");
// -->
```

index.html

```
...
<script type="text/javascript" src="myscript.js"></script>
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=******"></script>
...
```

Complete basic sample code:

```
<html>
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="chrome=1">
<title>Trying Watcheezy API</title>

// 0. The following script has to be placed before any line of Watcheezy API, it is then recommanded to place it at the beginning of the page.

<script type="text/javascript">

// the following parameters have to be created by the webmaster: the name of the parameters can be arbitrary, the value of the parameters has to be filled by your own algorithm/code, at the page loading. You can use whatever technology you what (this example use a little bit of PHP).

var cartValue =  <?php $cartAmount = getCartAmount($visitorId) ;  /* this is a fictive function of your own...*/ echo $cartAmount; /* result is 9 */ ?>
var nbProductInCart = 2;
var  visitorIsLogged = 1;
var  visitorIsUnderage = 0;
var  nbConnection = 99;
var visitorClass = "premium";
var nbPurchase = 3;
var  nbAbandon = 6;

var historicViews = ["airplane", "motorcycle", "rocket"];
var listVehicles = ["car", "bicycle"];
var historicVisits = ["http://www.watcheezy.com/contact.html"];
var listPages = ["http://www.watcheezy.com/contact.html", "http://www.watcheezy.com/cart.php"];

var visitorId = "John Smith";

// The following parameter defines your specific information to appear in Watcheezy Backoffice. Data must be classified as a JSON object. Each section is composed of a couple description/value which index are respectively 0 and 1
var visitorInfo = {
   "identity":
   {
      0: "customer reference",
      1: "5j3472-en",
   },
   "name":
   {
      0: "customer name",
      1: "John Smith"
   },
   "age":
   {
      0: "customer age",
      1 : 19
   },
   "historic":
   {
      0: "purchase history",
      1:
      [
         {
            0: "date",
            1: "2014-01-10"
         },
         {
            0: "product",
            1: "bike 5f48XZ"
         }
      ]
   }
};

</script>
```

```
</head>
<body>
<h1>API test</h1>
<p>Hello World</p>

// 1. The watcheezy main script has to be placed  first!
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=******"></script>

// 2. Now you can declare and use the API functions
<script type="text/javascript">

// a. Adding custom criteria for the visitors targeting

forceWatcheeAlert("interesting visitor"); // the visitor will trigger an alert, whatever happens ; the label of this alert will be "interesting visitor"

createRangeCriterion("There are between 2 and 5 products in the cart", "For this visitor, it triggers an alert if the number of product in his cart is
between 2 and 5", 2, 5, nbProductInCart ); //  if nbProductInCart = 2 none alert will be triggered
createRangeCriterion("Cart value is between 10$ and 20$", "...", 10, 20, cartValue ); // if  cartValue = 15 an alert will be triggered for this visitor

createBinaryCriterion("Visitor is logged", "...",  visitorIsLogged); // if  visitorIsLogged different from 0 an alert will be triggered for this visitor
createBinaryCriterion("Visitor is underage", "...", visitorIsUnderage); // if  visitorIsUnderage = 0 none alert will be triggered

createReferenceCriterion("Visitor 100th connection", "...", 100, nbConnection ); // if  nbConnection = 99 none alert will be triggered for this visitor
createReferenceCriterion("Visitor is premium", "...", "premium", visitorClass ); // if  visitorClass = "premium" an alert will be triggered for this visitor

createMinCriterion("Visitor gave up a minimum of 7 sales", "...", 7, nbAbandon) // if  nbAbandon > 6 an alert will be triggered for this visitor
createMinCriterion("Visitor had purchased at least 3 articles", "...", 3, nbPurchase) // if  nbPurchase < 3 none alert will be triggered

createSelectionCriterion("Visitor had load contact page or cart page", "...", listPages, historicVisits) ; /* if  any element of historicVisits is in
listPages, an alert is triggered*/
createSelectionCriterion("Visitor is watching an car or a bicycle", "...", listVehicles, historicViews) ; /*  if none element of historicViews is in
listVehicles none alert is triggered*/

// b. Adding information about your visitor. Those information belong to you (your database...)
watcheeSetCustomId(visitorId); // for this visitor, it will display "John Smith" in Watcheezy Trafic section
watcheeSetCustomInfo(visitorInfo); // for this visitor it will display all information given in visitorInfo parameter, in Watcheezy Trafic section

// c. Tab options
watcheeTabAutoOpen(); // the watcheezy tab is open directly at page loading
watcheeOpenTabOnTimeout(15); // the watcheezy service is loading after 15 seconds
//watcheeOpenTabOnCall() ; //if uncommented this function make the tab appear only if an agent is writing to the visitor

</script>
</body>
</html>
```

Advanced sample code:

In this sample, the watcheezy tab will appear under two conditions: 1. the cart amout is greater than 130$ and 2. the user is on the page for a minimum of 1 minute. Then an alert is triggered in the backoffice, displaying the customer name in the traffic section of Watcheezy.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>API Watcheezy - Advanced sample</title>
<script type="text/javascript" src="//www.watcheezy.net/deliver/watcheezy.js?key=********"></script>
<script type="text/javascript">

var cartAmout_js = 120; // 120 is given by yourself, however you want
var userName_js = "";

var cartThreshold_js = 130;
var msg_js = "cart amout is at least 130$";
var delay_js = 60;

</script>

</head>
<body>
   <p>Best umbrellas ever</p>

<script type="text/javascript">

<?php

// those session parameter are created and filled previously
$cartAmout_php = $_SESSION['cart_amout'];          // = 130$
$userIsLogged_php = $_SESSION['user_id'];          // = 123456789
$userName_php = $_SESSION['user_name'];            // = Jacob JOHNSON

?>

userName_js = "<?php echo $userName_php; ?>";
watcheeSetCustomId(userName_js); // the name of Jacob Jonhson will appear in Watcheezy trafic section, on the line that correspond to this visitor

cartAmout_js = "<?php echo $cartAmout_php; ?>";
createMinCriterion(msg_js, "", cartThreshold_js, cartAmout_js); // an alert will be triggered because the cart amout criteria is validated

if(cartAmout_js >= cartThreshold_js)
   watcheeOpenTabOnTimeout(delay_js); // the tab will be loaded after 60 seconds, only if the cart amout is greater than 130$ (which is the case here)
</script>
</body>
</html>
```

Combining different APIs:

When a visitor loads one of your page, it is possible to get additional data from external API, and reinject them into Watcheezy API. This case could happen if you have already shared visitor data with another (third party) API. Such APIs may provide additional data, like geolocation or gender... and you may be interesting in reinjecting thoses information into your Watcheezy backoffice.

Example: a well known external API provides “real time data” about online visitors (a third party “analytical” script is first placed on the page). One of the provided information is the geoposition of the visitor: the external API uses a “get” function and deliver a JSON. In this example, there is one visitor online, located in London (UK):

```
{
 "kind": "analytics#realtimeData",
 "id": "https://www.**********&dimensions=rt:city&metrics=rt:activeVisitors",
 "query": {
  "ids": "ga:########",
  "dimensions": "rt:city",
  "metrics": [
   "rt:activeVisitors"
  ],
  "max-results": 1000
 },
 "totalResults": 1,
 "selfLink": "https://www.*********&dimensions=rt:city&metrics=rt:activeVisitors",
 "profileInfo": {
  ...
 },
 "columnHeaders": [
  {
   "name": "rt:city",
   "columnType": "DIMENSION",
   "dataType": "STRING"
  },
  {
   "name": "rt:activeVisitors",
   "columnType": "METRIC",
   "dataType": "INTEGER"
  }
 ],
 "totalsForAllResults": {
  "rt:activeVisitors": "1"
 },
 "rows": [
  [
   "London",
   "1"
  ]
 ]
}
```

You may be interested in getting this “London” information and display it into your Watcheezy monitoring. All you have to do is to:

1. get this external data, first calling the external api

```
<?php

// link to the external API, like https://www.*******/analytics/v3/data/realtime?ids=ga%3######&metrics=rt%3AactiveVisitors&dimensions=rt
%3Acity&key={#####}

…

$results = $analytics->data_realtime->get('ga:#####', 'ga:activeVisitors', 'ga:city'); // request for one specific visitor ######

$rows = $results->getRows(); // parse request result

$city = htmlspecialchars($rows[0], ENT_NOQUOTES); // get the city for the unique visitor...

?>
```

2. reuse this parameter in watcheeSetCustomInfo function.

```
...
<script type="text/javascript">
var visitorInfo = {
    "location from G.A.":
    {
        0: "city",
        1: "<?php echo $city; ?>",
    }
};
</script>

<script type="text/javascript">
watcheeSetCustomInfo(visitorInfo);
</script>
...
```

List of functions

a. The following functions help you customizing your targeting strategy, adding criterion on the visitor. Each Criterion, if it is validated, trigger an alert in Watcheezy backoffice, in the Trafic setion.

createBinaryCriterion
createEqualityCriterion
createMaxCriterion
createMinCriterion
createRangeCriterion
createSelectionCriterion
createWatcheeAlert

a'. The following functions can be used asynchrone, to add a real time alert on a event of your choice.

createDynamicAlert
createCTRLDAlert

b. The following functions add custom information about the visitor. Those information are displayed in Watcheezy backoffice, in the Trafic section.They are available for this session only and not saved.

watcheeSetCustomId
watcheeSetCustomInfo

c. The following functions help you customizing tha oppening or the appearance of the Watcheezy tab.

watcheeHideTab
watcheeTabAutoOpen
watcheeOpenEvent
watcheeOpenTabOnCall
watcheeOpenTabOnEvent
watcheeOpenTabOnTimeout

d. The following functions allows to create special conditions, based on visitor historic. Those conditions are used as event to triggers Watcheezy openning functions (see section c.)
watcheeSetTag
watcheeAddKeyPage
watcheeDeleteKeyPage
watcheeIsKeyPageVisited
watcheeResetKeyPages

createBinaryCriterion

This function add a custom criterion to existing Watcheezy criteria.
If the criterion is validated, an alert is triggered in the Watcheezy Backoffice, in a special section.

To validate the criterion, this function test a value. If the value is different from zero, the criterion is validated ; if the value is equal to zero the criterion is not validated.

constructor
createBinaryCriterion(criterion_name, criterion_desc, current_value)

parameters

- criterion_name (string): short description of your criterion. This message will appear in the alert list as the cause of the alert. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).
- criterion_desc (string): complete description of your criterion. This value could be a empty string. Maximum length is 150 characters (otherwise, the string is truncated to 150 letters).
- current_value (integer/string): this parameter is tested once at the time of the function call. If it is set to zero, none alert is trigger, if it is different from zero, an alert is triggered. The zero value could be a string "0" or an integer 0. null and false values are considered as zero values.

returns
Boolean: Returns true on success

example of use

This function could be used to test if a user is logged:
In your server-side there must be a session parameter containing the id of the user, and this id is different from zero. So this id could be passed as an argument to the function in the client-side, then resulting an alert in Watcheezy backoffice.
Ex.1:

```
var id = "" ;
...
id= "azerty1234" ; // this value is set by your server-side at page loading
…
var msg = "visitor is logged" ;
createBinaryCriterion(msg, "", id); // an alert will be triggered. Label for this alert is "visitor is logged"
```

Ex.2:

```
var  id= "";
...
id = <?php echo 0 ;?> // because your server did not identifyed the user
...
var msg = "visitor is logged" ;
createBinaryCriterion(msg, "", id); // none alert will be triggered.
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
You can add as many of these kind of criterion, then calling as much function in the source code.
The fact that a criterion triggers an alert in the backoffice (or not) has nothing to do with the return value of the function.

createEqualityCriterion

This function add a custom criterion to existing Watcheezy criteria.
If the criterion is validated, an alert is triggered in the Watcheezy Backoffice, in a special section.

To validate the criterion, this function test a value. If this value is equal a reference value (= = =), the criterion is validated ; if the tested value is different from the reference value (!= =), the criterion is not validated.

constructor
createEqualityCriterion(criterion_name, criterion_desc, reference_value, current_value)

parameters

- criterion_name (string): short description of your criterion. This message will appear in the alert list as the cause of the alert. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).
- criterion_desc (string): complete description of your criterion. This value could be a empty string. Maximum length is 150 characters (otherwise, the string is truncated to 150 letters).
- reference_value (integer/float/string): the value for witch the current_value must be equal to trigger an alert.
- current_value (integer/float/string): this parameter is tested once at the time of the function call. If it is strictly equal to reference_value, an alert is triggered.

returns
Boolean: Returns true on success

example of use

This function could be used to mark a visitor depending on its zipcode or a keyword on the page.
In your server-side there must be a session parameter containing the adress of the registred visitor. An alert is triggered in Watcheezy backoffice if this number correspond to the agent location.
Ex.1:

```
var agentZipCode = 75009; // if the agent is from Paris IX
var zipCode = 0 ;
...
zipCode= 75009; // this value is set by your server-side at page loading
…
var msg = "visitor is closes to your location" ;
createEqualityCriterion(msg, "", agentZipCode , zipCode); // an alert will be triggered. Label inWatcheezy  is "visitor is closes to your location"
```

Ex.2:

```
var pageKeyWord = "faq"; // if an alert must be triggered for each visitor who visits the FAQ page
var currentPageKeyWord = "" ;
...
currentPageKeyWord = "faq" ; // this value is set by your server-side at page loading
…
var msg = "visitor is looking for help" ;
createEqualityCriterion(msg, "", pageKeyWord , currentPageKeyWord); // an alert will be triggered
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
You can add as many of these kind of criterion, then calling as much function in the source code.
The fact that a criterion triggers an alert in the backoffice (or not) has nothing to do with the return value of the function.
The type of current_value and reference_value are tested. For instance; 1234 is different from "1234"
This test is case sensitive.

createMaxCriterion

This function add a custom criterion to existing Watcheezy criteria.
If the criterion is validated, an alert is triggered in the Watcheezy Backoffice, in a special section.

To validate the criterion, this function test a value. If this value is below a threshold value (<=), the criterion is validated ; if the tested value is strictly greater than the threshold value (>), the criterion is not validated.

constructor
createMaxCriterion(criterion_name, criterion_desc, max_value, current_value)

parameters

- criterion_name (string): short description of your criterion. This message will appear in the alert list as the cause of the alert. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).
- criterion_desc (string): complete description of your criterion. This value could be a empty string. Maximum length is 150 characters (otherwise, the string is truncated to 150 letters).
- max_value (integer/float): it represent a threshold: the maximum value for witch the current_value must not exceed to trigger an alert.
- current_value (integer/float): this parameter is tested once at the time of the function call. If it is lesser or equal to max_value, an alert is trigger. If it is stricly superior to max_value , none alert occurs for this criterion.

returns
Boolean: Returns true on success

example of use

This function could be used to mark a visitor until he have put n products in its cart.
In your server-side there must be a session parameter containing the number of product in the user's cart. Unless this number is n, an alert is triggered in Watcheezy backoffice, so that an agent can interpell the user to convince him.
Ex.1:

```
var maxNbProduct = 2; // visitor has to be convinced until he has put 2 products in his cart
var nbProductInCart = 0 ;
...
nbProductInCart= "1" ; // this value is set by your server-side at page loading
…
var msg = "less than two product in the cart" ;
createLimitCriterion(msg, "", maxNbProduct  nbProductInCart); // an alert will be triggered. Label for this alert is "less than two product in the cart"
```

Ex.2:

```
var maxNbProduct = 5; // visitor has to be convinced until he has put 5 products in his cart
var nbProductInCart = 0 ;
...
nbProductInCart= "4" ; // this value is set by your server-side at page loading
…
var msg = "less than 5 product in the cart" ;
createLimitCriterion(msg, "", maxNbProduct , nbProductInCart); // none alert will be triggered.
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
You can add as many of these kind of criterion, then calling as much function in the source code.
The fact that a criterion triggers an alert in the backoffice (or not) has nothing to do with the return value of the function.
Do not to be confused with the function createMinCriterion which triggers an alert from a specific threshold.

createMinCriterion

This function add a custom criterion to existing Watcheezy criteria.
If the criterion is validated, an alert is triggered in the Watcheezy Backoffice, in a special section.

This function is quite the oposite to createMaxCriterion. It creates an alert from a minimum value.

To validate the criterion, this function test a value. If this value is greater or equal a threshold value (>=), the criterion is validated ; if the tested value is below the threshold value (<), the criterion is not validated.

constructor
createMinCriterion(criterion_name, criterion_desc, min_value, current_value)

parameters

- criterion_name (string): short description of your criterion. This message will appear in the alert list as the cause of the alert. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).
- criterion_desc (string): complete description of your criterion. This value could be a empty string. Maximum length is 150 characters (otherwise, the string is truncated to 150 letters).
- min_value (integer/float): it represent a threshold: the minimum value for witch the current_value must exceed to trigger an alert.
- current_value (integer/float): this parameter is tested once at the time of the function call. If it is greater or equal to max_value, an alert is triggered. If it is stricly inferior to min_value, none alert occurs for this criterion.

returns
Boolean: Returns true on success

example of use

This function could be used to mark a visitor when he made a number of purchase.
In your server-side there must be a session parameter containing the number of product the user's bought. An alert is triggered in Watcheezy backoffice when this number reaches a certain value, so that an agent can help this VIP user

```
var minNbSales = 7; // if a visitor have already bought 7 products, is is considered as VIP
var nbPastSales = 0 ;
...
nbPastSales= "9" ; // this value is set by your server-side at page loading
…
var msg = "good customer: help as much as possible" ;
createMinCriterion(msg, "", minNbSales , nbProductInCart); // an alert will be triggered. Label inWatcheezy  is "good customer... possible"
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
You can add as many of these kind of criterion, then calling as much function in the source code.
The fact that a criterion triggers an alert in the backoffice (or not) has nothing to do with the return value of the function.
Do not to be confused with the function createMaxCriterion which triggers an alert bellow a specific threshold.

createRangeCriterion

This function add a custom criterion to existing Watcheezy criteria.
If the criterion is validated, an alert is triggered in the Watcheezy Backoffice, in a special section.

It creates an alert from a range of values: two conditions have to be satisfied.

To validate the criterion, this function test a value. If this value is greater or equal a threshold value (>=), the criterion is validated ; if the tested value is below the threshold value (<), the criterion is not validated.

constructor
createRangeCriterion(criterion_name, criterion_desc, min_value, max_value, current_value)

parameters

- criterion_name (string): short description of your criterion. This message will appear in the alert list as the cause of the alert. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).
- criterion_desc (string): complete description of your criterion. This value could be a empty string. Maximum length is 150 characters (otherwise, the string is truncated to 150 letters).
- min_value (integer/float): it represent the first threshold: the minimum value that the current_value must exceed to trigger an alert.
- max_value (integer/float): it represent the second threshold: the maxmum value that the current_value must not exceed to trigger an alert.
- current_value (integer/float): this parameter is tested once at the time of the function call. If it is greater or equal to min_value AND lesser or equal to max_value, an alert is triggered. Otherwise, none alert occurs for this criterion.

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used to mark a visitor when he made a number of purchase.
In your server-side there must be a session parameter containing the number of product the user's bought. An alert is triggered in Watcheezy backoffice when this number reaches a certain value, so that an agent can help this VIP user

```
var minCartAmout = 10; // if the visitor's cart contain at least 10$
var maxCartAmout = 90; // if the visitor's does not exceed 90$
var cartAmout = 0 ;
...
cartAmout= "26" ; // this value is set by your server-side at page loading
…
var msg = "The cart amout is between "+ minCartAmout  + "$ and " + maxCartAmout + "$";
createRangeCriterion(msg, "", minCartAmout, maxCartAmout , cartAmout); // an alert will be triggered.
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
You can add as many of these kind of criterion, then calling as much function in the source code.
The fact that a criterion triggers an alert in the backoffice (or not) has nothing to do with the return value of the function.

createSelectionCriterion

This function add a custom criterion to existing Watcheezy criteria.
If the criterion is validated, an alert is triggered in the Watcheezy Backoffice, in a special section.

It creates an alert from a list of values.

To validate the criterion, this function test two arrays. If a value in the candidate array is found in the reference array, the criterion is validated ; if none element of the candidate array if found, the criterion is not validated.

constructor
createSelectionCriterion(criterion_name, criterion_desc, reference_array, candidate_array)

parameters

- criterion_name (string): short description of your criterion. This message will appear in the alert list as the cause of the alert. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).
- criterion_desc (string): complete description of your criterion. This value could be a empty string. Maximum length is 150 characters (otherwise, the string is truncated to 150 letters).
- reference_array (array of integer/float/string): is composed of value that has to be found.
- candidate_array (array of integer/float/string): this parameter is tested once at the time of the function call. If at least one element of this array is found in reference_array, an alert is triggered. Otherwise, none alert occurs for this criterion. The array can be composed of a unique element.

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used to mark a visitor when he visits a specific page/product among a list of predefined pages/products, like support or contact pages.
In your server-side there may be a session parameter containing the historic of the user. An alert is triggered in Watcheezy backoffice if your personal historic (for this visitor) contain an url that correspond to your support or contact features.

```
var listPages = ["http://www.watcheezy.com/contact.html","http://www.watcheezy.com/support.php"];
var historicOfVisits = [ ];
...
historicOfVisits= ["http://www.watcheezy.com/contact.html"]; // this value is set by your server-side at page loading
…
var msg = "User is trying to contact the support";
createSelectionCriterion(msg, "",  listPages, historicOfVisits); // an alert will be triggered.
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
You can add as many of these kind of criterion, then calling as much function in the source code.
The fact that a criterion triggers an alert in the backoffice (or not) has nothing to do with the return value of the function.

createWatcheeAlert

When this function is called, it creates an alert in Watcheezy backoffice. This alert is associated with the visitor that load the page. The alert is effective for 30 minutes.

constructor
createWatcheeAlert(msg)

parameters

- msg (string): short description of the alert. This message will appear in the alert list in Watcheezy trafic section. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used to mark a visitor. The conditions of calling this function are all yours: it could be any of previous examples.

```
var myMoodIsFine = 0;
var msg = "User is interesting";
…
myMoodIsFine = 1; // this value is set by your own
...
if( myMoodIsGood == 1)
   createWatcheeAlert(msg); // an alert will be triggered.
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
You can add as many of these kind of alert, then calling as much function in the source code.

createDynamicAlert

When this function is called, it creates an alert in Watcheezy backoffice. This alert is associated with the visitor that load the page.
This function can be call asynchronous during visitor surf. It could be, for example, mapped to a button or a javascipt event.
The alert is effective for 30 minutes for the visitor.

constructor
createDynamicAlert(msg)

parameters

- msg (string): short description of the alert. This message will appear in the alert list in Watcheezy trafic section. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used when a visitor put a product in his cart. In that case the function could be associated to a onclick event.

```
...
var productNameOnThisPage = "bicycleXB48s";
</script>
…
<input type="button" onclick="createDynamicAlert(productNameOnThisPage+' has just been added to cart !');" value="add to cart">
...
```

notes
This function has to be called after the watcheezy api loading.
You can add as many of these kind of alert, then calling as much function in the source code.
The full Watcheezy service must be loaded before the function is active: the main library watcheezy.js must be completely loaded. Also it is useless to call this function at page loading, you should use createWatcheeAlert instead.

createCTRLDAlert

When this function is called, it creates an alert in Watcheezy backoffice, when the visitor press keyboard combination CTRL+D, which is the classical shortcut for faving/bookmarking a page. It is not yet possible to detect that a visitor bookmarks a page using another way (in the context menu or drag-droping an url).
This function can be call asynchronous during visitor surf. It should be associated to a javascript onkeydown event.

constructor
createCTRLDAlert(msg, _event_)

parameters

- msg (string): short description of the alert. This message will appear in the alert list in Watcheezy trafic section. We recommand to link this message to bookmark action. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).
- _event_: event object that contain the keyboard data. This parameter could be let at '_event_' (see example bellow)

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used when a visitor bookmark the current page using CTRL and D keys at the same time. In that case the function could be associated to a onkeydown event.

```
...
document.onkeydown = function(_event_){createCTRLDAlert('page is bookmarked', _event_)};
...
```

notes
This function has to be called after the watcheezy api loading.
The full Watcheezy service must be loaded before the function is active: the main library watcheezy.js must be completely loaded.

watcheeSetCustomId

This function replaces the default identity of a visitor in Watcheezy backoffice, trafic section, for the visitor that load the page. This identity could be a name, an id or whatever that helps the agent to recognize the visitor in the list.

constructor
watcheeSetCustomId(customId)

parameters

- customId (string): short description of the visitor. This message will appear in the visitors list in Watcheezy trafic section. Maximum length is 60 characters (otherwise, the string is truncated to 60 letters).

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used to identify a visitor, with data of your own.

```
var visitorName = "John Smith"; // this value is set by your own, for ex your database or login procedure
...
watcheeSetCustomId(visitorName); // "John Smith" will appear in Watcheezy backoffice
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
This function has to be called once.

WatcheeDynamicCustomId

This function replaces the default identity of a visitor in Watcheezy backoffice, trafic section, for the visitor that load the page. This identity could be a name, an id or whatever that helps the agent to recognize the visitor in the list.
This function can be call asynchronous. It could be, for example, mapped to a button, a javascipt event or a callback.
It is important that Watcheezy main script is already loaded before calling this function :
A try-catch method may be used : *try{watcheeDynamicCustomInfo(...)}catch(err){}*
A typeof test may be used too : *if(typeof(watcheeDynamicCustomInfo) == 'function') watcheeDynamicCustomInfo(...)*

constructor
see watcheeSetCustomId function

parameters
see watcheeSetCustomId function

returns
see watcheeSetCustomId function

example of use

This function could be used to identify a visitor, with data of your own.

```
var visitorName = "John Smith"; // this value is set by your own, for ex your database or login procedure
...
$( "p" ).click(function(){
  watcheeSetCustomId(visitorName); //assume that jquery is used
});
```

notes
This function has to be called after the watcheezy api loading, and after the watcheezy main script.
Each time the function is called, the data about the visitor are overwritten.

watcheeSetCustomInfo

This function add data about the visitor that load the page, in the Watcheezy Trafic section of the backoffice.
The data will be converted as a intelligible tab linked to the visitor default information.

constructor
watcheeSetCustomInfo(customInfo)

parameters

- customInfo (string(json)): your data about the visitor. The string is formated as a specific JSON. All information must be dual [description + value] and for each of those couple the json key is [0,1] (see the example bellow)

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used to identify a visitor, with data of your own.

```
var visitorInfo = {
            "identity":                  // coding purpose // won't be displayed
            {
               0: "customer reference",  // this label will be displayed // must be indexed as 0
               1: "5j3472-en"            // this value will be displayed // must be indexed to 1
            },
            "name":
            {
               0: "customer name",       // comma at the end of the line
               1: "John Smith"           // no comma ending the line
            },                           // comma ending the line
            "age":
            {
               0: "customer age",
               1 : 19
            },
            "historic":                  // same structure for arrays
            {
               0: "purchase history",    // label that will be displayed
               1:                        // describe the array hereafter
               [
                  {                      // first value
                     0: "date",          // first label // displayed // index is 0
                     1: "2014-01-10"     // first associated value // displayed // index is 1
                  },
                  {                      // second value
                     0: "product",
                     1: "bike 5f48XZ"
                  }
               ]
            }                            // no comma
};
...
watcheeSetCustomInfo(visitorInfo); // those data will appear in Watcheezy backoffice
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
This function has to be called once.

watcheeDynamicCustomInfo

This function add data about the visitor that load the page, in the Watcheezy Trafic section of the backoffice.
The data will be converted as a intelligible tab linked to the visitor default information.
This function can be call asynchronous. It could be, for example, mapped to a button, a javascipt event or a callback.
It is important that Watcheezy main script is already loaded before calling this function :
A try-catch method may be used : *try{watcheeDynamicCustomInfo(...)}catch(err){}*
A typeof test may be used too : *if(typeof(watcheeDynamicCustomInfo) == 'function') watcheeDynamicCustomInfo(...)*

constructor
see watcheeSetCustomInfo function

parameters
see watcheeSetCustomInfo function

returns
see watcheeSetCustomInfo function

example of use

This function could be used to identify a visitor, with data of your own.

```
var visitorInfo = {
                "identity":                         // coding purpose // won't be displayed
                {
                   0: "customer reference",    // this label will be displayed // must be indexed as 0
                   1: "5j3472-en"                 // this value will be displayed // must be indexed to 1
                }
};
…
$( "p" ).click(function(){
  watcheeDynamicCustomInfo(visitorInfo); //assume that jquery is used
});
```

notes
This function has to be called after the watcheezy api loading, and after the watcheezy main script.
Each time the function is called, the data about the visitor are overwritten.

watcheeHideTab

This function defines a specific condition for Watcheezy tab.
If called, the Watcheezy tab is not displayed for this page : both visitor and agents won't be able to communicate.
However, the tracking is still active for this visit: even if the Watcheezy tab is not visible, the visitor is still recorded for this visit.
Statistics provided in the Backoffice take those visits into account.

constructor
watcheeHideTab()

parameters
none

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used to get the real state of traffic on your pages, even on page where you don't want the Watcheezy tab to appear.

```
var thisPageIsIntereting = "";
…
thisPageIsIntereting = true; // set by your own code at page loading
…
if(thisPageIsIntereting)
   watcheeHideTab(); // the visitor can not see Watcheezy tab, but its visit is taken into account for your statistics.
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
This function has to be called once.

watcheeTabAutoOpen

This function defines a specific condition for Watcheezy tab.
If called, the Watcheezy tab is open directly: it is no more need to click on the tab to make it opened. The list of agent is then visible directly.
When the tab is open, user only have to click on the tab title to make the tab closing.

constructor
watcheeTabAutoOpen()

parameters
none

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used on pages where the visitor needs to contact an agent, with a direct understanding of the Watcheezy tab use (ex: for support purpose)

```
var tabNeedsToBeOpen = "";
…
tabNeedsToBeOpen = true; // set by your own code at page loading
…
if(tabNeedsToBeOpen)
   watcheeTabAutoOpen(); // the visitor see Watcheezy tab open directly
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
This function has to be called once.

watcheeOpenTabOnCall

This function defines a specific condition for Watcheezy tab.
If called, the Watcheezy tab is loaded but not visible at page loading. The full service appears automatically (and only) when an agent write a chat masseage to the visitor. From this moment, for the visitor, the Watchezy service remains open as normally. If he reloads the page, Watcheezy will be hidden again (until an agent write to him...). The use of this function cancel any previous watcheeHideTab or watcheeOpenTabOnEvent functions.
Tracking functionalities remains active even if the tab is not visible.

constructor
watcheeOpenTabOnCall()

parameters
none

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used on pages where the user may not need to ask question to agent, but the agent may want to help him (in product page for instance).

```
var productPage = "";
…
productPage = "bicycle"; // set by your own code at page loading
…
if(productPage !=  "contact" && productPage !=  "purchase") // the visitor can not see Watcheezy tab as he is not on "contact" or "purchase" pages;
   watcheeOpenTabOnCall();  // but its visit is seen by agents that can interpell him.
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
This function has to be called once.
This function has no effect if watcheeHideTab is called.
This function is active only after the eventual delay induced by watcheeOpenTabOnTimeout. If a call is made during this period, the tab watcheezy will not open.

watcheeOpenTabOnEvent

This function defines a custom condition for Watcheezy tab.
If called, the Watcheezy tab is loaded but not visible at page loading. The full service appears automatically (and only) when the function watcheeOpenEvent is called. From this moment, for the visitor, the Watchezy service remains open as normally. If he reloads the page, Watcheezy will be hidden again (until watcheeOpenEvent is called again...).
Tracking functionalities remains active even if the tab is not visible. The use of this function cancel any previous watcheeHideTab or watcheeOpenTabOnCall functions.

constructor
watcheeOpenTabOnEvent()

parameters
none

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used to triggers the watcheezy service opening when the visitor press a button (supportive / help purpose)

```
var productPage = "";
…
productPage = "bicycle"; // set by your own code at page loading
…
if(productPage ==  "bicycle" && productPage ==  "motorbike") // the visitor can display the Watcheezy in product page;
   watcheeOpenTabOnEvent();  // The tab is hidden until watcheeOpenEvent is called (see below).
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
The Watcheezy service must be completely loaded to use this function (it has no effect otherwise)
This function has no effect if watcheeHideTab is called.
This function applies watcheeOpenTabOnCall functionnality: if an agent interpells the visitor, the Watcheezy tab opens.
This function is active only after the eventual delay induced by watcheeOpenTabOnTimeout. I**f a call is made during this period, the tab watcheezy will not open.**

watcheeOpenEvent

This function, combined with watcheeOpenTabOnEvent, forces the display of Watcheezy tab.
From this moment, for the visitor, the Watchezy service remains open as normally. If he reloads the page, Watcheezy will be hidden again (until watcheeOpenEvent is called again...). It has no effect if the function watcheeOpenTabOnEvent is not used beforehand.
Tracking functionalities remains active even if the tab is not visible.

constructor
watcheeOpenEvent()

parameters
none

returns
Boolean: Returns true on success. Returns false otherwise (if watcheezy is not fully loaded for example)

example of use

This function could be used to triggers the watcheezy service opening when the visitor press a button.

```
...
 watcheeOpenTabOnEvent();  // previously defined (between watcheezy_api.js and watcheezy.js main script)
</script>
…
<input type="button" onclick="watcheeOpenEvent();" value="I need help"> // if the visitor clicks on this button, the Watcheezy tab appears
...
```

notes
This function has to be called after the watcheezy api loading.
This function can be called dynamically, and mapped to a javascript event (onclick...)
The Watcheezy service must be completely loaded to use this function (it has no effect otherwise)
This function has no effect if watcheeHideTab is called.
WatcheeOpenTabOnEvent must be called at page loading, otherwise this function has no effect.

watcheeOpenTabOnTimeout

This function defines a specific condition for Watcheezy tab.
If called, the Watcheezy tab is displayed after a given delay. At the end of this countdown, the tab appears automatically (in reduced mode). From this moment, for the visitor, the Watchezy service remains open as normally. If he reloads the page, Watcheezy will be delayed again.

constructor
watcheeOpenTabOnTimeout(delay)

parameters

- delay (integer): duration, in seconds, while the service is not loaded. This value must be between 0 and 300 (5 minutes).

returns
Boolean: Returns true on success. Returns false otherwise.

example of use

This function could be used if agents are busy and there is no need to make the watcheezy appear immediatly.

```
var agentsAreBusy = "";
var agentsAreBusy = 120;
…
agentsAreBusy = true; // set by your own code at page loading
…
if(agentsAreBusy)
   watcheeOpenTabOnTimeout(delay); // the visitor can not see Watcheezy tab for 2 minutes, time for the agent to be available again.
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
This function has to be called once.
This function has no effect if watcheeHideTab is called

watcheeSetTag

This function tag a specific page.
The tags could be used to select a speific agent in the chat, or trigger a specific tagretting rule

constructor
watcheeSetTag(keyword)

parameters

- keyword (string): a keyword that represent the page that is beeing visited. This could be a single word a keyword representing the role of the current page.

returns
Boolean: Returns true on success. Returns false otherwise (if cookie can not be written).

example of use

This function could be used if the visitor is visiting an important page (like Bosh/product/drill).

```
…
watcheeSetTag('dishwasher'); // the tag is associated with this page
//and
watcheeSetTag('product');
//and
watcheeSetTag('indesit'); /
...
```

notes
This function can be called for each keyword you want to be saved.

watcheeAddKeyPage

This function add a specific information (keyword) on the visitor's browser.
It creates a cookie in the visitor browser that contain the keyword. This cookie is available for 30 minutes, that correspond to a classical 'unique visitor' session.
This function may be used in collaboration with watcheeIsKeyPageVisited, to trigger a specific event like openning the Watcheeze tab.

constructor
watcheeAddKeyPage(keyword)

parameters

- keyword (string): a keyword that represent the page that is beeing visited. This could be an url or just a keyword representing the role of the current page.

returns
Boolean: Returns true on success. Returns false otherwise (if cookie can not be written).

example of use

This function could be used if the visitor is visiting an important page (like cart/shipping/paying) and you want this information to be re-used if the visitor visits some other pages (typically if he does not valid a purchase).

```
…
<meta name="keywords" content="shipping">
<title>Shipping information</title>
…
watcheeAddKeyPage('shipping'); // a cookie is created with the value 'shipping'
//and/or
watcheeAddKeyPage('http://www.watcheezy.com/shipping.html'); /
...
// see watcheeIsKeyPageVisited for the rest of the example
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
This function can be called for each keyword you want to be saved.
Keywords are unique, no duplicate is possible.

watcheeDeleteKeyPage

This function remove a specific information (keyword) from the visitor historis. This keyword is potentially written by watcheeAddKeyPage in visitor's browser: it explore the specific cookie in the visitor's browser and remove the keyword.

constructor
watcheeDeleteKeyPage(keyword)

parameters

- keyword (string): a keyword that represent a page that has been visited. This could be an url or just a keyword representing the role of the current page.

returns
Boolean: Returns true on success. Returns false otherwise (if cookie can not be written).

example of use

This function could be used if keywords had been set by watcheeAddKeyPage in previous pages, and is now useless.

Any previous page :

```
…
watcheeAddKeyPage('shipping');
watcheeAddKeyPage('command'); // the cookie contain two new keywords (but eventually many others)  'shipping' and 'command'
…
```

current page :

```
…
watcheeDeleteKeyPage('command'); // now the cookie does not contain the keyword 'command' any more
…
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.
This function should be called for each keyword you want to delete.

watcheeIsKeyPageVisited

This function test if the current keyword had been previously recorded. It tests the value among the cookies eventually created on visitor's browser.
It is mainly is used in a control statement ('if' condition) to trigger special event (opening Watcheezy tab)

constructor
watcheeIsKeyPageVisited(keyword)

parameters

- keyword (string): keyword that is tested

returns
Boolean: Returns true if current keyword belong to the list of saved keywords.
Returns false otherwise: if the keyword does not belong to the existing list, or there is no more cookie in the visitor browser (>30min) or cookies are not allowed.

example of use

This function could be used if you want the Watcheezy tab to appear only if the visitor has visited an important page (like cart/shipping/paying).

```
…
if(!watcheeIsKeyPageVisited('shipping'))
{
   watcheeHideTab(); // the tab won't be loaded until the visitor has visit the 'shipping' page
}
...
```

it is possible to combine many functions with this test:

```
…
if(watcheeIsKeyPageVisited('purchase'))
{
   watcheeOpenTabOnTimeout(10); // tab will open after 30 seconds (only if 'purchase' page had been visited)
}
else
   watcheeHideTab(); // otherwise the tab won't be displayed
...
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.

watcheeResetKeyPages

This function remove all specific information (keywords) from the visitor's browser. Those keywords had been potentially written by watcheeAddKeyPage.
It delete the cookie that stocks pages keywords.

constructor
watcheeResetKeyPages()

parameters
none

returns
Boolean: Returns true on success. Returns false otherwise (no access to cookies)

example of use

This function could be used to manually reset the keywords. For example if a user has just purchased or bought a product, and you want its historic to be reset as a newcomer visitor.

```
…
watcheeResetKeyPages();
...
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.

watcheeSetLead

This function creates a lead at the page load. Data will by display on the visitor card. Data must come from the webmaster. This new lead is then adeed to the other leads of your watcheezy account.

constructor
watcheeSetLead(name, coord, comment);

parameters
name (string): identity of the user
coord (string): contact information of the user
comment (string): information about the lead (whatever you want)

returns
Boolean: Returns true on success. Returns false otherwise (no access to cookies)

example of use

```
...
var  name = <?php echo $mycrm.name; ?> // $mycrm.name is "John" for example
var coord = "main street";
var comment = "is interested by...";
watcheeSetLead(name, coord, comment);
...
```

notes
This function has to be called after the watcheezy api loading, and before the watcheezy main script.