

```

.ui-autocomplete-loading { background: white
url("images/M_images/ui-anim_basic_16x16.gif") right center no-repeat; } /** * vkBeautify -
javascript plugin to pretty-print or minify text in XML, JSON, CSS and SQL formats. * *
Version - 0.99.00.beta * Copyright (c) 2012 Vadim Kiryukhin * vkiryukhin @ gmail.com *
http://www.eslinstructor.net/vkbeautify/ * * MIT license: *
http://www.opensource.org/licenses/mit-license.php * * Pretty print * *
vkbeautify.xml(text [,indent_pattern]); * vkbeautify.json(text [,indent_pattern]); *
vkbeautify.css(text [,indent_pattern]); * vkbeautify.sql(text [,indent_pattern]); * * @text
- String; text to beautify; * @indent_pattern - Integer | String; * Integer: number of
white spaces; * String: character string to visualize indentation ( can also be a set of
white spaces ) * Minify * * vkbeautify.xmlmin(text [,preserve_comments]); *
vkbeautify.jsonmin(text); * vkbeautify.cssmin(text [,preserve_comments]); *
vkbeautify.sqlmin(text); * * @text - String; text to minify; * @preserve_comments -
Bool; [optional]; * Set this flag to true to prevent removing comments from @text (
minxml and mincss functions only. ) * * Examples: * vkbeautify.xml(text); // pretty print
XML * vkbeautify.json(text, 4 ); // pretty print JSON * vkbeautify.css(text, ' . . . '); //
pretty print CSS * vkbeautify.sql(text, '----'); // pretty print SQL * *
vkbeautify.xmlmin(text, true); // minify XML, preserve comments * vkbeautify.jsonmin(text); //
minify JSON * vkbeautify.cssmin(text); // minify CSS, remove comments ( default ) *
vkbeautify.sqlmin(text); // minify SQL * */ (function() { function createShiftArr(step) { var
space = ' '; if ( isNaN(parseInt(step)) ) { // argument is string space = step; } else { //
argument is integer switch(step) { case 1: space = ' '; break; case 2: space = ' '; break;
case 3: space = ' '; break; case 4: space = ' '; break; case 5: space = ' '; break;
case 6: space = ' '; break; case 7: space = ' '; break; case 8: space = ' '; break;
case 9: space = ' '; break; case 10: space = ' '; break; case 11: space = '
'; break; case 12: space = ' '; break; } } var shift = ['\n']; // array of shifts
for(ix=0;ixs{0,} -1 || ar[ix].search(/>/) > -1 || ar[ix].search(/!DOCTYPE/) > -1 ) { inComment =
false; } } else // end comment or // if(ar[ix].search(/-->/) > -1 || ar[ix].search(/>/) >
-1) { str += ar[ix]; inComment = false; } else // // if( /<

```