HTML5 Tutorial

[HTML Introduction](https://www.w3schools.com/html/html_intro.asp)

[HTML Editors](https://www.w3schools.com/html/html_editors.asp)

[HTML Basic](https://www.w3schools.com/html/html_basic.asp)

[HTML Elements](https://www.w3schools.com/html/html_elements.asp)

[HTML Attributes](https://www.w3schools.com/html/html_attributes.asp)

[HTML Headings](https://www.w3schools.com/html/html_headings.asp)

[HTML Paragraphs](https://www.w3schools.com/html/html_paragraphs.asp)

[HTML Styles](https://www.w3schools.com/html/html_styles.asp)

[HTML Formatting](https://www.w3schools.com/html/html_formatting.asp)

[HTML Quotations](https://www.w3schools.com/html/html_quotation_elements.asp)

[HTML Comments](https://www.w3schools.com/html/html_comments.asp)

[HTML Colors](https://www.w3schools.com/html/html_colors.asp)

[HTML CSS](https://www.w3schools.com/html/html_css.asp)

[HTML Links](https://www.w3schools.com/html/html_links.asp)

[HTML Images](https://www.w3schools.com/html/html_images.asp)

[HTML Tables](https://www.w3schools.com/html/html_tables.asp)

[HTML Lists](https://www.w3schools.com/html/html_lists.asp)

[HTML Blocks](https://www.w3schools.com/html/html_blocks.asp)

[HTML Classes](https://www.w3schools.com/html/html_classes.asp)

[HTML Iframes](https://www.w3schools.com/html/html_iframe.asp)

[HTML JavaScript](https://www.w3schools.com/html/html_scripts.asp)

[HTML File Paths](https://www.w3schools.com/html/html_filepaths.asp)

[HTML Head](https://www.w3schools.com/html/html_head.asp)

[HTML Layout](https://www.w3schools.com/html/html_layout.asp)

[HTML Responsive](https://www.w3schools.com/html/html_responsive.asp)

[HTML Computercode](https://www.w3schools.com/html/html_computercode_elements.asp)

[HTML Entities](https://www.w3schools.com/html/html_entities.asp)

[HTML Symbols](https://www.w3schools.com/html/html_symbols.asp)

[HTML Charset](https://www.w3schools.com/html/html_charset.asp)

[HTML URL Encode](https://www.w3schools.com/html/html_urlencode.asp)

[HTML XHTML](https://www.w3schools.com/html/html_xhtml.asp)

[HTML Forms](https://www.w3schools.com/html/html_forms.asp)

[HTML Form Elements](https://www.w3schools.com/html/html_form_elements.asp)

[HTML Input Types](https://www.w3schools.com/html/html_form_input_types.asp)

[HTML Input Attributes](https://www.w3schools.com/html/html_form_attributes.asp)

[HTML5 Intro](https://www.w3schools.com/html/html5_intro.asp)

[HTML5 Support](https://www.w3schools.com/html/html5_browsers.asp)

[HTML5 New Elements](https://www.w3schools.com/html/html5_new_elements.asp)

[HTML5 Semantics](https://www.w3schools.com/html/html5_semantic_elements.asp)

[HTML5 Migration](https://www.w3schools.com/html/html5_migration.asp)

[HTML5 Style Guide](https://www.w3schools.com/html/html5_syntax.asp)

[HTML Canvas](https://www.w3schools.com/html/html5_canvas.asp)

[HTML SVG](https://www.w3schools.com/html/html5_svg.asp)

[HTML Google Maps](https://www.w3schools.com/html/html_googlemaps.asp)

HTML Media

[HTML Video](https://www.w3schools.com/html/html5_video.asp)

[HTML Audio](https://www.w3schools.com/html/html5_audio.asp)

[HTML Plug-ins](https://www.w3schools.com/html/html_object.asp)

[HTML YouTube](https://www.w3schools.com/html/html_youtube.asp)

HTML APIs

[HTML Geolocation](https://www.w3schools.com/html/html5_geolocation.asp)

[HTML Drag/Drop](https://www.w3schools.com/html/html5_draganddrop.asp)

[HTML Local Storage](https://www.w3schools.com/html/html5_webstorage.asp)

[HTML Web Workers](https://www.w3schools.com/html/html5_webworkers.asp)

[HTML SSE](https://www.w3schools.com/html/html5_serversentevents.asp)

----------------------------------------------------------------------------------------------

CSS Tutorial

[CSS HOME](https://www.w3schools.com/css/default.asp)

[CSS Introduction](https://www.w3schools.com/css/css_intro.asp)

[CSS Syntax](https://www.w3schools.com/css/css_syntax.asp)

[CSS How To](https://www.w3schools.com/css/css_howto.asp)

[CSS Colors](https://www.w3schools.com/css/css_colors.asp)

[CSS Backgrounds](https://www.w3schools.com/css/css_background.asp)

[CSS Borders](https://www.w3schools.com/css/css_border.asp)

[CSS Margins](https://www.w3schools.com/css/css_margin.asp)

[CSS Padding](https://www.w3schools.com/css/css_padding.asp)

[CSS Height/Width](https://www.w3schools.com/css/css_dimension.asp)

[CSS Box Model](https://www.w3schools.com/css/css_boxmodel.asp)

[CSS Outline](https://www.w3schools.com/css/css_outline.asp)

[CSS Text](https://www.w3schools.com/css/css_text.asp)

[CSS Fonts](https://www.w3schools.com/css/css_font.asp)

[CSS Icons](https://www.w3schools.com/css/css_icons.asp)

[CSS Links](https://www.w3schools.com/css/css_link.asp)

[CSS Lists](https://www.w3schools.com/css/css_list.asp)

[CSS Tables](https://www.w3schools.com/css/css_table.asp)

[CSS Display](https://www.w3schools.com/css/css_display_visibility.asp)

[CSS Max-width](https://www.w3schools.com/css/css_max-width.asp)

[CSS Position](https://www.w3schools.com/css/css_positioning.asp)

[CSS Overflow](https://www.w3schools.com/css/css_overflow.asp)

[CSS Float](https://www.w3schools.com/css/css_float.asp)

[CSS Inline-block](https://www.w3schools.com/css/css_inline-block.asp)

[CSS Align](https://www.w3schools.com/css/css_align.asp)

[CSS Combinators](https://www.w3schools.com/css/css_combinators.asp)

[CSS Pseudo-class](https://www.w3schools.com/css/css_pseudo_classes.asp)

[CSS Pseudo-element](https://www.w3schools.com/css/css_pseudo_elements.asp)

[CSS Opacity](https://www.w3schools.com/css/css_image_transparency.asp)

[CSS Navigation Bar](https://www.w3schools.com/css/css_navbar.asp)

[CSS Dropdowns](https://www.w3schools.com/css/css_dropdowns.asp)

[CSS Tooltips](https://www.w3schools.com/css/css_tooltip.asp)

[CSS Image Gallery](https://www.w3schools.com/css/css_image_gallery.asp)

[CSS Image Sprites](https://www.w3schools.com/css/css_image_sprites.asp)

[CSS Attr Selectors](https://www.w3schools.com/css/css_attribute_selectors.asp)

[CSS Forms](https://www.w3schools.com/css/css_form.asp)

[CSS Counters](https://www.w3schools.com/css/css_counters.asp)

[CSS3 Introduction](https://www.w3schools.com/css/css3_intro.asp)

[CSS3 Rounded Corners](https://www.w3schools.com/css/css3_borders.asp)

[CSS3 Border Images](https://www.w3schools.com/css/css3_border_images.asp)

[CSS3 Backgrounds](https://www.w3schools.com/css/css3_backgrounds.asp)

[CSS3 Colors](https://www.w3schools.com/css/css3_colors.asp)

[CSS3 Gradients](https://www.w3schools.com/css/css3_gradients.asp)

[CSS3 Shadows](https://www.w3schools.com/css/css3_shadows.asp)

[CSS3 Text](https://www.w3schools.com/css/css3_text_effects.asp)

[CSS3 Fonts](https://www.w3schools.com/css/css3_fonts.asp)

[CSS3 2D Transforms](https://www.w3schools.com/css/css3_2dtransforms.asp)

[CSS3 3D Transforms](https://www.w3schools.com/css/css3_3dtransforms.asp)

[CSS3 Transitions](https://www.w3schools.com/css/css3_transitions.asp)

[CSS3 Animations](https://www.w3schools.com/css/css3_animations.asp)

[CSS3 Images](https://www.w3schools.com/css/css3_images.asp)

[CSS3 Buttons](https://www.w3schools.com/css/css3_buttons.asp)

[CSS3 Pagination](https://www.w3schools.com/css/css3_pagination.asp)

[CSS3 Multiple Columns](https://www.w3schools.com/css/css3_multiple_columns.asp)

[CSS3 User Interface](https://www.w3schools.com/css/css3_user_interface.asp)

[CSS3 Box Sizing](https://www.w3schools.com/css/css3_box-sizing.asp)

[CSS3 Flexbox](https://www.w3schools.com/css/css3_flexbox.asp)

[CSS3 Media Queries](https://www.w3schools.com/css/css3_mediaqueries.asp)

[CSS3 MQ Examples](https://www.w3schools.com/css/css3_mediaqueries_ex.asp)

CSS Responsive

[RWD Intro](https://www.w3schools.com/css/css_rwd_intro.asp)

[RWD Viewport](https://www.w3schools.com/css/css_rwd_viewport.asp)

[RWD Grid View](https://www.w3schools.com/css/css_rwd_grid.asp)

[RWD Media Queries](https://www.w3schools.com/css/css_rwd_mediaqueries.asp)

[RWD Images](https://www.w3schools.com/css/css_rwd_images.asp)

[RWD Videos](https://www.w3schools.com/css/css_rwd_videos.asp)

[RWD Frameworks](https://www.w3schools.com/css/css_rwd_frameworks.asp)

[RWD Templates](https://www.w3schools.com/css/css_rwd_templates.asp)

CSS Sorax

CSS 01 Введение

CSS 02 Основные понятия

CSS 03 Селекторы

CSS 04 Позиционирование

CSS 05 Текст

CSS 06 Цвет

CSS 07 Рамки

CSS 08 border-radius

CSS 09 Треугольники

CSS 10 Фон

CSS 11 Спрайты

CSS 12 float и clearfix

CSS 13 Списки

CSS 14 box-shadow

CSS 15 transition

CSS 16 Градиенты

CSS 17 transform

CSS 18 Трехмерный куб

CSS 19 animation

CSS 20 @font-face

CSS 21 media queries

CSS 22 Верстка сайта

CSS 23 Многоколоночный текст

----------------------------------------------------------------------------------------------

Bootstrap Tutorial

[BS HOME](https://www.w3schools.com/bootstrap/default.asp)

[BS Get Started](https://www.w3schools.com/bootstrap/bootstrap_get_started.asp)

[BS Grid Basic](https://www.w3schools.com/bootstrap/bootstrap_grid_basic.asp)

[BS Typography](https://www.w3schools.com/bootstrap/bootstrap_typography.asp)

[BS Tables](https://www.w3schools.com/bootstrap/bootstrap_tables.asp)

[BS Images](https://www.w3schools.com/bootstrap/bootstrap_images.asp)

[BS Jumbotron](https://www.w3schools.com/bootstrap/bootstrap_jumbotron_header.asp)

[BS Wells](https://www.w3schools.com/bootstrap/bootstrap_wells.asp)

[BS Alerts](https://www.w3schools.com/bootstrap/bootstrap_alerts.asp)

[BS Buttons](https://www.w3schools.com/bootstrap/bootstrap_buttons.asp)

[BS Button Groups](https://www.w3schools.com/bootstrap/bootstrap_button_groups.asp)

[BS Glyphicons](https://www.w3schools.com/bootstrap/bootstrap_glyphicons.asp)

[BS Badges/Labels](https://www.w3schools.com/bootstrap/bootstrap_badges_labels.asp)

[BS Progress Bars](https://www.w3schools.com/bootstrap/bootstrap_progressbars.asp)

[BS Pagination](https://www.w3schools.com/bootstrap/bootstrap_pagination.asp)

[BS Pager](https://www.w3schools.com/bootstrap/bootstrap_pager.asp)

[BS List Groups](https://www.w3schools.com/bootstrap/bootstrap_list_groups.asp)

[BS Panels](https://www.w3schools.com/bootstrap/bootstrap_panels.asp)

[BS Dropdowns](https://www.w3schools.com/bootstrap/bootstrap_dropdowns.asp)

[BS Collapse](https://www.w3schools.com/bootstrap/bootstrap_collapse.asp)

[BS Tabs/Pills](https://www.w3schools.com/bootstrap/bootstrap_tabs_pills.asp)

[BS Navbar](https://www.w3schools.com/bootstrap/bootstrap_navbar.asp)

[BS Forms](https://www.w3schools.com/bootstrap/bootstrap_forms.asp)

[BS Inputs](https://www.w3schools.com/bootstrap/bootstrap_forms_inputs.asp)

[BS Inputs 2](https://www.w3schools.com/bootstrap/bootstrap_forms_inputs2.asp)

[BS Input Sizing](https://www.w3schools.com/bootstrap/bootstrap_forms_sizing.asp)

[BS Media Objects](https://www.w3schools.com/bootstrap/bootstrap_media_objects.asp)

[BS Carousel](https://www.w3schools.com/bootstrap/bootstrap_carousel.asp)

[BS Modal](https://www.w3schools.com/bootstrap/bootstrap_modal.asp)

[BS Tooltip](https://www.w3schools.com/bootstrap/bootstrap_tooltip.asp)

[BS Popover](https://www.w3schools.com/bootstrap/bootstrap_popover.asp)

[BS Scrollspy](https://www.w3schools.com/bootstrap/bootstrap_scrollspy.asp)

[BS Affix](https://www.w3schools.com/bootstrap/bootstrap_affix.asp)

Bootstrap Grids

[BS Grid System](https://www.w3schools.com/bootstrap/bootstrap_grid_system.asp)

[BS Stacked/Horizontal](https://www.w3schools.com/bootstrap/bootstrap_grid_stacked_to_horizontal.asp)

[BS Grid Small](https://www.w3schools.com/bootstrap/bootstrap_grid_small.asp)

[BS Grid Medium](https://www.w3schools.com/bootstrap/bootstrap_grid_medium.asp)

[BS Grid Large](https://www.w3schools.com/bootstrap/bootstrap_grid_large.asp)

[BS Grid Examples](https://www.w3schools.com/bootstrap/bootstrap_grid_examples.asp)

Bootstrap Themes

[BS Templates](https://www.w3schools.com/bootstrap/bootstrap_templates.asp)

[BS Theme "Simply Me"](https://www.w3schools.com/bootstrap/bootstrap_theme_me.asp)

[BS Theme "Company"](https://www.w3schools.com/bootstrap/bootstrap_theme_company.asp)

[BS Theme "Band"](https://www.w3schools.com/bootstrap/bootstrap_theme_band.asp)

Bootstrap Exam

[BS Examples](https://www.w3schools.com/bootstrap/bootstrap_examples.asp)[BS Quiz](https://www.w3schools.com/bootstrap/bootstrap_quiz.asp)[BS Certificate](https://www.w3schools.com/bootstrap/bootstrap_exam.asp)

Bootstrap CSS Ref

[CSS All Classes](https://www.w3schools.com/bootstrap/bootstrap_ref_all_classes.asp)

[CSS Typography](https://www.w3schools.com/bootstrap/bootstrap_ref_css_text.asp)

[CSS Buttons](https://www.w3schools.com/bootstrap/bootstrap_ref_css_buttons.asp)

[CSS Forms](https://www.w3schools.com/bootstrap/bootstrap_ref_css_forms.asp)

[CSS Helpers](https://www.w3schools.com/bootstrap/bootstrap_ref_css_helpers.asp)

[CSS Images](https://www.w3schools.com/bootstrap/bootstrap_ref_css_images.asp)

[CSS Tables](https://www.w3schools.com/bootstrap/bootstrap_ref_css_tables.asp)

[CSS Dropdowns](https://www.w3schools.com/bootstrap/bootstrap_ref_comp_dropdowns.asp)

[CSS Navs](https://www.w3schools.com/bootstrap/bootstrap_ref_comp_navs.asp)[Glyphicons](https://www.w3schools.com/bootstrap/bootstrap_ref_comp_glyphs.asp)

Bootstrap JS Ref

[JS Affix](https://www.w3schools.com/bootstrap/bootstrap_ref_js_affix.asp)

[JS Alert](https://www.w3schools.com/bootstrap/bootstrap_ref_js_alert.asp)

[JS Button](https://www.w3schools.com/bootstrap/bootstrap_ref_js_button.asp)

[JS Carousel](https://www.w3schools.com/bootstrap/bootstrap_ref_js_carousel.asp)

[JS Collapse](https://www.w3schools.com/bootstrap/bootstrap_ref_js_collapse.asp)

[JS Dropdown](https://www.w3schools.com/bootstrap/bootstrap_ref_js_dropdown.asp)

[JS Modal](https://www.w3schools.com/bootstrap/bootstrap_ref_js_modal.asp)

[JS Popover](https://www.w3schools.com/bootstrap/bootstrap_ref_js_popover.asp)

[JS Scrollspy](https://www.w3schools.com/bootstrap/bootstrap_ref_js_scrollspy.asp)

[JS Tab](https://www.w3schools.com/bootstrap/bootstrap_ref_js_tab.asp)

[JS Tooltip](https://www.w3schools.com/bootstrap/bootstrap_ref_js_tooltip.asp)

----------------------------------------------------------------------------------------------

JS Tutorial

[JS HOME](https://www.w3schools.com/js/default.asp)

[JS Introduction](https://www.w3schools.com/js/js_intro.asp)

[JS Where To](https://www.w3schools.com/js/js_whereto.asp)

[JS Output](https://www.w3schools.com/js/js_output.asp)

[JS Syntax](https://www.w3schools.com/js/js_syntax.asp)

[JS Statements](https://www.w3schools.com/js/js_statements.asp)

[JS Comments](https://www.w3schools.com/js/js_comments.asp)

[JS Variables](https://www.w3schools.com/js/js_variables.asp)

[JS Operators](https://www.w3schools.com/js/js_operators.asp)

[JS Arithmetic](https://www.w3schools.com/js/js_arithmetic.asp)

[JS Assignment](https://www.w3schools.com/js/js_assignment.asp)

[JS Data Types](https://www.w3schools.com/js/js_datatypes.asp)

[JS Functions](https://www.w3schools.com/js/js_functions.asp)

[JS Objects](https://www.w3schools.com/js/js_objects.asp)

[JS Scope](https://www.w3schools.com/js/js_scope.asp)

[JS Events](https://www.w3schools.com/js/js_events.asp)

[JS Strings](https://www.w3schools.com/js/js_strings.asp)

[JS String Methods](https://www.w3schools.com/js/js_string_methods.asp)

[JS Numbers](https://www.w3schools.com/js/js_numbers.asp)

[JS Number Methods](https://www.w3schools.com/js/js_number_methods.asp)

[JS Math](https://www.w3schools.com/js/js_math.asp)

[JS Random](https://www.w3schools.com/js/js_random.asp)

[JS Dates](https://www.w3schools.com/js/js_dates.asp)

[JS Date Formats](https://www.w3schools.com/js/js_date_formats.asp)

[JS Date Methods](https://www.w3schools.com/js/js_date_methods.asp)

[JS Arrays](https://www.w3schools.com/js/js_arrays.asp)

[JS Array Methods](https://www.w3schools.com/js/js_array_methods.asp)

[JS Array Sort](https://www.w3schools.com/js/js_array_sort.asp)

[JS Booleans](https://www.w3schools.com/js/js_booleans.asp)

[JS Comparisons](https://www.w3schools.com/js/js_comparisons.asp)

[JS Conditions](https://www.w3schools.com/js/js_if_else.asp)

[JS Switch](https://www.w3schools.com/js/js_switch.asp)

[JS Loop For](https://www.w3schools.com/js/js_loop_for.asp)

[JS Loop While](https://www.w3schools.com/js/js_loop_while.asp)

[JS Break](https://www.w3schools.com/js/js_break.asp)

[JS Type Conversion](https://www.w3schools.com/js/js_type_conversion.asp)

[JS Bitwise](https://www.w3schools.com/js/js_bitwise.asp)

[JS RegExp](https://www.w3schools.com/js/js_regexp.asp)

[JS Errors](https://www.w3schools.com/js/js_errors.asp)

[JS Debugging](https://www.w3schools.com/js/js_debugging.asp)

[JS Hoisting](https://www.w3schools.com/js/js_hoisting.asp)

[JS Strict Mode](https://www.w3schools.com/js/js_strict.asp)

[JS Style Guide](https://www.w3schools.com/js/js_conventions.asp)

[JS Best Practices](https://www.w3schools.com/js/js_best_practices.asp)

[JS Mistakes](https://www.w3schools.com/js/js_mistakes.asp)

[JS Performance](https://www.w3schools.com/js/js_performance.asp)

[JS Reserved Words](https://www.w3schools.com/js/js_reserved.asp)

[JS Versions](https://www.w3schools.com/js/js_versions.asp)

[JS JSON](https://www.w3schools.com/js/js_json.asp)

JS Forms

[JS Forms](https://www.w3schools.com/js/js_validation.asp)[Forms API](https://www.w3schools.com/js/js_validation_api.asp)

JS Objects

[Object Definitions](https://www.w3schools.com/js/js_object_definition.asp)

[Object Properties](https://www.w3schools.com/js/js_properties.asp)

[Object Methods](https://www.w3schools.com/js/js_object_methods.asp)

[Object Prototypes](https://www.w3schools.com/js/js_object_prototypes.asp)

JS Functions

[Function Definitions](https://www.w3schools.com/js/js_function_definition.asp)

[Function Parameters](https://www.w3schools.com/js/js_function_parameters.asp)

[Function Invocation](https://www.w3schools.com/js/js_function_invocation.asp)

[Function Call](https://www.w3schools.com/js/js_function_call.asp)

[Function Apply](https://www.w3schools.com/js/js_function_apply.asp)

[Function Closures](https://www.w3schools.com/js/js_function_closures.asp)

JS HTML DOM

[DOM Intro](https://www.w3schools.com/js/js_htmldom.asp)

[DOM Methods](https://www.w3schools.com/js/js_htmldom_methods.asp)

[DOM Document](https://www.w3schools.com/js/js_htmldom_document.asp)

[DOM Elements](https://www.w3schools.com/js/js_htmldom_elements.asp)

[DOM HTML](https://www.w3schools.com/js/js_htmldom_html.asp)

[DOM CSS](https://www.w3schools.com/js/js_htmldom_css.asp)

[DOM Animations](https://www.w3schools.com/js/js_htmldom_animate.asp)

[DOM Events](https://www.w3schools.com/js/js_htmldom_events.asp)

[DOM Event Listener](https://www.w3schools.com/js/js_htmldom_eventlistener.asp)

[DOM Navigation](https://www.w3schools.com/js/js_htmldom_navigation.asp)

[DOM Nodes](https://www.w3schools.com/js/js_htmldom_nodes.asp)

[DOM Collections](https://www.w3schools.com/js/js_htmldom_collections.asp)

[DOM Node Lists](https://www.w3schools.com/js/js_htmldom_nodelist.asp)

JS Browser BOM

[JS Window](https://www.w3schools.com/js/js_window.asp)

[JS Screen](https://www.w3schools.com/js/js_window_screen.asp)

[JS Location](https://www.w3schools.com/js/js_window_location.asp)

[JS History](https://www.w3schools.com/js/js_window_history.asp)

[JS Navigator](https://www.w3schools.com/js/js_window_navigator.asp)

[JS Popup Alert](https://www.w3schools.com/js/js_popup.asp)

[JS Timing](https://www.w3schools.com/js/js_timing.asp)

[JS Cookies](https://www.w3schools.com/js/js_cookies.asp)

JS AJAX

[AJAX Intro](https://www.w3schools.com/js/js_ajax_intro.asp)

[AJAX XMLHttp](https://www.w3schools.com/js/js_ajax_http.asp)

[AJAX Request](https://www.w3schools.com/js/js_ajax_http_send.asp)

[AJAX Response](https://www.w3schools.com/js/js_ajax_http_response.asp)

[AJAX XML File](https://www.w3schools.com/js/js_ajax_xmlfile.asp)

[AJAX PHP](https://www.w3schools.com/js/js_ajax_php.asp)

[AJAX ASP](https://www.w3schools.com/js/js_ajax_asp.asp)

[AJAX Database](https://www.w3schools.com/js/js_ajax_database.asp)

[AJAX Applications](https://www.w3schools.com/js/js_ajax_applications.asp)

[AJAX Examples](https://www.w3schools.com/js/js_ajax_examples.asp)

JS JSON

[JSON Intro](https://www.w3schools.com/js/js_json_intro.asp)

[JSON Syntax](https://www.w3schools.com/js/js_json_syntax.asp)

[JSON vs XML](https://www.w3schools.com/js/js_json_xml.asp)

[JSON Data Types](https://www.w3schools.com/js/js_json_datatypes.asp)

[JSON Objects](https://www.w3schools.com/js/js_json_objects.asp)

[JSON Arrays](https://www.w3schools.com/js/js_json_arrays.asp)

[JSON Parse](https://www.w3schools.com/js/js_json_parse.asp)

[JSON Stringify](https://www.w3schools.com/js/js_json_stringify.asp)

[JSON PHP](https://www.w3schools.com/js/js_json_php.asp)

[JSON HTML](https://www.w3schools.com/js/js_json_html.asp)

[JSON JSONP](https://www.w3schools.com/js/js_json_jsonp.asp)

JS Sorax

JS 01 Введение

JS 02 Инструкции

JS 03 Выражения и операторы  
JS 04 Типы данных и

переменные  
JS 05 Числа  
JS 06 Строки  
JS 07 Логические значения  
JS 08 null и undefined  
JS 09 Преобразование типов  
JS 10 Условные инструкции  
JS 11 Условный оператор  
JS 12 Циклы  
JS 13 Функции

JS 14 Цепочки областей

видимости  
JS 15 Замыкания  
JS 16 Обработка исключений  
JS 17 Объекты  
JS 18 this и непрямой вызов

методов  
JS 19 Аксессоры и атрибуты

свойств  
JS 20 Прототипы и

наследование  
JS 21 Конструкторы и классы  
JS 22 Цепные вызовы методов  
JS 23 J50N  
JS 24 Массивы  
JS 25 ЕS5 методы массивов  
JS 26 Объект Math  
JS 27 Класс Date  
JS 28 Регулярные выражения  
JS 29 Класс RegExp  
JS 30 Strict Mode  
JS 31 Выполнение JS в

браузере

JS 32 Window

JS 33 Location

JS 34 History, Screen,

Navigator  
JS 35 DOM

----------------------------------------------------------------------------------------------

jQuery Tutorial

[jQuery HOME](https://www.w3schools.com/jquery/default.asp)

[jQuery Intro](https://www.w3schools.com/jquery/jquery_intro.asp)

[jQuery Get Started](https://www.w3schools.com/jquery/jquery_get_started.asp)

[jQuery Syntax](https://www.w3schools.com/jquery/jquery_syntax.asp)

[jQuery Selectors](https://www.w3schools.com/jquery/jquery_selectors.asp)

[jQuery Events](https://www.w3schools.com/jquery/jquery_events.asp)

jQuery Effects

[jQuery Hide/Show](https://www.w3schools.com/jquery/jquery_hide_show.asp)

[jQuery Fade](https://www.w3schools.com/jquery/jquery_fade.asp)

[jQuery Slide](https://www.w3schools.com/jquery/jquery_slide.asp)

[jQuery Animate](https://www.w3schools.com/jquery/jquery_animate.asp)

[jQuery stop()](https://www.w3schools.com/jquery/jquery_stop.asp)

[jQuery Callback](https://www.w3schools.com/jquery/jquery_callback.asp)

[jQuery Chaining](https://www.w3schools.com/jquery/jquery_chaining.asp)

jQuery HTML

[jQuery Get](https://www.w3schools.com/jquery/jquery_dom_get.asp)

[jQuery Set](https://www.w3schools.com/jquery/jquery_dom_set.asp)

[jQuery Add](https://www.w3schools.com/jquery/jquery_dom_add.asp)

[jQuery Remove](https://www.w3schools.com/jquery/jquery_dom_remove.asp)

[jQuery CSS Classes](https://www.w3schools.com/jquery/jquery_css_classes.asp)

[jQuery css()](https://www.w3schools.com/jquery/jquery_css.asp)

[jQuery Dimensions](https://www.w3schools.com/jquery/jquery_dimensions.asp)

jQuery Traversing

[jQuery Traversing](https://www.w3schools.com/jquery/jquery_traversing.asp)

[jQuery Ancestors](https://www.w3schools.com/jquery/jquery_traversing_ancestors.asp)

[jQuery Descendants](https://www.w3schools.com/jquery/jquery_traversing_descendants.asp)

[jQuery Siblings](https://www.w3schools.com/jquery/jquery_traversing_siblings.asp)

[jQuery Filtering](https://www.w3schools.com/jquery/jquery_traversing_filtering.asp)

jQuery AJAX

[jQuery AJAX Intro](https://www.w3schools.com/jquery/jquery_ajax_intro.asp)

[jQuery Load](https://www.w3schools.com/jquery/jquery_ajax_load.asp)

[jQuery Get/Post](https://www.w3schools.com/jquery/jquery_ajax_get_post.asp)

jQuery Misc

[jQuery noConflict()](https://www.w3schools.com/jquery/jquery_noconflict.asp)

----------------------------------------------------------------------------------------------

**курси prometheus Основи Web UI розробки**

var myMassage; оголошення змінної

myMassage = "hello, user"; присвоєння змінній значення

alert(myMassage); виведення значення змінної на екран через діалогове вікно

усі ідентифікатори змінних РегІсТРозаЛеЖнІ

у присвоєнні значень змінних ' та " ровнозначні:

myMassage = "hello, user"; = myMassage = 'hello, user';

змінні можуть складатися із букв, цифр, \_ та $, змінні не можуть починатися з цифри, у якості змінної не можна викоистовувати зарезервовані слова

Типи даних:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | number | число (ціле або дробове) | var a = 12.34; |
| 2 | string | рядок | myMassage = "hello, user"; або myMassage = 'hello, user'; |
| 3 | boolean | логічний тип | true або false (пишуться без лапок) |
| 4 | null | спец.значення | пишеться без лапок |
| 5 | undefined | спец.значення | з'явиться, якщо alert(оголошена змінна без присвоєного значення) |
|  | infinity | безкінечність | Браузер виведе її напр. у випадку ділення на нуль. |
|  | NaN | не число | Браузер виведе його, напр. якщо до числа додати рядок. Формально належить до типу "число". |
| 6 | Об'єкти | назва = {тип: значення}; | напр. user = {name: "Vasja"}; |

ВЗАЄМОДІЯ ІЗ КОРИСТУВАЧЕМ

|  |  |
| --- | --- |
| alert('you massage here'); | виведення повідомлення у модальне вікно |
| var result = prompt('як тебе звати?', 'василь'); | виводиться модальне вікно із запитанням, формою для відповіді (в ній буде запропоновано 'василь', кнопками ок та скасувати). Якщо зробити так: var result = prompt('як тебе звати?'); , то форма буде порожня. Якщо натиснути скасувати - змінна result = null, якщо ввести ім'я та натиснути ок - змінна result = ім'я. |
| var result = confirm('ти повнолітній?'); | виводиться модальне вікно із запитанням, кнопками ок та скасувати). Якщо натиснути скасувати - змінна result = false, якщо натиснути ок - змінна result = true. |

УМОВИ

var year = +prompt('When Javascript first appeared?', '');

if (year === 1995) {

alert('You are right!!!');

} else {

alert('I"m disappointed...');

}

(+prompt означає перетворення відповіді користувача у число)

або:

var year = +prompt('When Javascript first appeared?');

if (year === 1995) {

alert('You are right!!!');

} else if(1990 < year && year < 2000){

alert('You were close...');

} else {

alert('I"m disappointed...');

}

довга форма запису умови:

if (age > 18) {

access = true;

} else {

access = false;

}

коротка форма запису умови:

access = (age > 18) ? true : false;

ЦИКЛИ

var i = 0;

while (i < 3) {

alert(i):

i++;

}

var i = 0;

do {

alert(i);

i++;

} while (i < 3);

for (var i = 0; i < 3; i++) {

alert(i);

}

var i = 0;

for (; i < 3; i++) {

alert(i);

}

переривання вічного циклу:

while (true) { /\* умова вічного циклу \*/

var value = prompt('Enter number:', '');

if (value == null) { /\* або if (!value) break; \*/

break;

}

alert('Your number is: ' + value);

}

пропуск ітерації в циклі:

for виконується у послідовності: умова-тіло-крок.

for(var i = 0; i < 4; i++) {

if (i === 2) {

continue; /\* для парних if (i % 2 == 0) continue;\*/

}

alert(i);

}

(i > 5) ? alert(i) : continue; /\* код буде помилковим, бо синтаксичні конструкції, які не повертають значень (наприклад break/continue), не можна використовувати з оператором ? \*/

конструкція switch - case - break (для заміни багатьох else if):

var value = +prompt('2 + 2 =', '');

switch(value) {

case 3:

alert('More...');

break;

case 4:

alert('Great!');

break;

case 5:

alert('Less...');

break;

default: /\* default - для решти значень \*/

alert('I don"t know this value');

}

дана конструкція замінює:

var value = +prompt('2 + 2 =', '');

if (value === 3) {

alert('More...');

} else if (value === 4) {

alert('Great!');

} else if (value === 5) {

alert('Less...'); } else {

alert('I don"t know this value')

}

ФУНКЦІЇ

1. function declaration.

function count (num1, num2, num3) {

var sum = num1 + num2 + num3;

console.log(sum);

}

count (5, 3, 2); /\*виклик функції із указанням параметрів\*/

function count (num1, num2, num3) {

var num1 = num1 || 1; /\* якщо змінна не буде передана ззовні,

var num2 = num2 || 2; буде використано дефолтне значення 1 \*/

var num3 = num3 || 3;

var sum = num1 + num2 + num3;

console.log(sum);

}

count (10, 10); /\* результат - 23 \*/

count (10); /\* результат - 15 \*/

Звернення до функції можливе перед її оголошенням

count (10);

function count (num1, num2, num3) {

var sum = num1 + num2 + num3;

console.log(sum);

}

2. function expression.

var count = function (num1, num2, num3) {

var num1 = num1 || 1;

var num2 = num2 || 2;

var num3 = num3 || 3;

var sum = num1 + num2 + num3;

console.log(sum);

}

count (10, 10);

У такому випадку звернення до функції перед її оголошенням не можливе.

Такий спосіб оголошення функцій використовується за наявності умов (в середині конструкції if, оскільки function declaration в if не спрацює):

var age = prompt('Скільки тобі років?');

var sayHi;

if (age >= 18) {

sayHi = function() {

alert( 'Прошу Вас!' );

}

} else {

sayHi = function() {

alert( 'До 18 не можна!' );

}

}

sayHi()

3. анонімні функції.

(function (num1, num2, num3) {

var num1 = num1 || 1;

var num2 = num2 || 2;

var num3 = num3 || 3;

var sum = num1 + num2 + num3;

console.log(sum);

})();

У такому випадку ім'я не обов'язкове, але може бути присвоєне, якщо виклик функції відбувається з тіла самої функції - рекурсивний виклик:

(function count (num1, num2, num3) { /\* приклад вічної рекурсії \*/

var num1 = num1 || 1;

var num2 = num2 || 2;

var num3 = num3 || 3;

var sum = num1 + num2 + num3;

console.log(sum);

count();

})();

іще приклад:

function ask(question, yes, no) {

if (confirm(question)) yes()

else no();

}

ask(

"Ви згодні?", /\* question \*/

function() { alert("Ви погодились."); }, /\* yes приклад анонімної функції \*/

function() { alert("Ви скасували виконання."); } /\* no приклад анонімної функції \*/

);

4. new function.

var f = new function('x,y', 'return x + y');

console.log (f(1, 2));

-----

приклад рекурсії:

function pow(x, n) {

if (n != 1) {

return x \* pow(x, n - 1);

} else {

return x;

}

}

alert( pow(2, 3) ); // 8

будь-яка рекурсія може бути перероблена у цикл і це буде більш ефективно в плані використання ресурсів машини:

function pow(x, n) {

var result = x;

for (var i = 1; i < n; i++) {

result \*= x;

}

return result;

}

|  |  |
| --- | --- |
| var id = document.getElementById("id\_name"); | змінна id, яка вміщує у себе елемент з ідентифікатором id\_name |
| var class = document.getElementsByClassName("class\_name"); | змінна (масив) class, яка вміщує у себе усі елементи з класом class\_name |
| var tagArr = document.getElementsByTagName("p"); | змінна (масив) tagArr, яка вміщує у себе усі елементи з тегом p |
| var elements = document.querySelectorAll("input"); | змінна (масив) elements, яка вміщує у себе усі input-елементи |
| var element = document.querySelector("input"); | змінна element, яка вміщує у себе перший input-елемент |
| var value = document.querySelector("input[type='text']").value; | змінна value, якій присвоєне значення першого input-елементу з типом text |
| var child = document.createElement(li); | створення нового тегу li |
| var text = document.createTextNode('trololo'); | створення тексту trololo |
| child.Elem.appendChild (text); | додавання тексту trololo в кінець елементу child |
| var parent = document.getElementById("list");  var liItems = document.getElevtntByTagName("li");  parent.insertBefore(child, liItems[1]); | додавання елементу child в масив parent перед другим елементом масиву liItems |
| var parent = document.getElementById("list");  var liItems = document.getElevtntByTagName("li");  parent.removeChild(liItems[0]); | видалення першого елементу масиву |
| parent.replaceChild(newItem, oldItem); | заміна одного елементу на інший |

ПОДІЇ.

<h1 id="title" onclick="this.innerHTML = 'Oooops!'">Click on this text</h1> заміна тексту при кліку по ньому

те саме, тільки з javascript:

1. html: <h1 id="title" onclick="changeText(this)">Click on this text</h1>

js: function changeText(element) {

element.innerHTML = "Oooops!";

};

2. html: <h1 id="title" onclick="changeText(this)">Click on this text</h1>

js: function changeText(element) {

document.getElementById("title").innerHTML = "Oooops!";

};

3. html: <h1 id="title">Click on this text</h1>

js: document.getElementById("title").onclick = function() {

this.innerHTML = "Oooops!";

};

найчастіше вживані події:

onclick

onload завантаження документу або певної його частини

onchange зміна стану чекбоксів та радіобаттонів

onmouseover, onmouseout проведення мишею над об'єктом

onmousedown перша частина onclick: натиснення кнопки миші

onmouseup друга частина onclick: відпускання кнопки миші

onkeypress, onkeyup, onkeydown те саме, що і onclick, тільки з клавіатурою

onfocus, onblur у фокусі / не у фокусі

learn.javascript.ru

defer/async

Під час виявлення <script async src="..."> браузер не зупиняє обробку сторінки, скрипт виконається тільки тоді, коли буде завантажений. З defer відбудеться теж саме, але є кілька відмінностей:

з async першим виконається скрипт, який першим завантажиться, а при defer скрипти виконуватимуться по порядку.

async-скрипт виконається одразу, як тільки завантажиться, defer-скрипт виконається тільки після того, як завантажиться уся сторінка.

При одночасному використанні defer/async буде використаний async.

defer/async є атрибутами тільки для зовнішніх скриптів.

Якщо тег <script> був доданий через javascript, то він себе поводитиме як async. Якщо потрібно зберегти порядок відтворення скриптів, потрібно використати властивість script.async = false.

-----

"use strict";

Цей рядок вставляється на початок скрипта, якщо потрібно, щоб скрипт відповідав сучасним вимогам ES5. "use strict"; також можна вказувати на початку функцій, тоді він працюватиме тільки в середині функції.

-----

typeof x;

використовується для повернення типу змінної х.

typeof null // "object" – офіційно визнана помилка.

typeof function(){} // "function" – функції є об’єктами, але так зручніше.

-----

var c = 3 - (a = b + 1); оператор "=" повертає значення

-----

Скорочена арифметика з присвоюванням:

n += 5; = n = n + 5;

n \*= 2; = n = n \* 2;

-----

Значення null і undefined рівні одне одному і не рівні нічому іншому.

При перетворенні у число null стає 0, а undefined стає NaN.

alert( null > 0 ); // false

alert( null == 0 ); // false

alert(null >= 0); // true

alert( undefined > 0 ); // false

alert( undefined < 0 ); // false

alert( undefined == 0 ); // false

-----

Оператор if перетворює число 0, порожній рядок, null, undefined та NaN у false, усе інше – в true.

if (0) { // false, умова невиконувана

...

}

if (1) { // true, умова виконувана

...

}

Кілька операторів ?

var age = prompt('возраст?', 18);

var message = (age < 3) ? 'Здравствуй, малыш!' :

(age < 18) ? 'Привет!' :

(age < 100) ? 'Здравствуйте!' :

'Какой необычный возраст!';

alert( message );

-----

Так можна, але краще не треба:

var company = prompt('Какая компания создала JavaScript?', '');

(company == 'Netscape') ?

alert('Да, верно') : alert('Неправильно');

-----

Логічні оператори || (АБО), && (ТА) і ! (НЕ).

JavaScript вираховує кілька АБО зліва направо. Якщо він знаходить хоча б один true, то наступні елементи ігноруються. Якщо усі елементи false, то JavaScript повертає останнє false-значення:

var x;

true || (x = 1);

alert(x); // undefined, x не присвоєно

alert( undefined || 0 ); // 0

JavaScript вираховує кілька ТА зліва направо. Якщо він знаходить хоча б один false, він його повертає, усі наступні елементи ігноруються. Якщо усі елементи true, то JavaScript повертає останнє true-значення:

alert( 1 && 2 && null && 3 ); // null

alert( 1 && 2 && 3 ); // 3

&&(ТА) має вищий пріоритет за || (АБО).

alert( 5 || 1 && 0 ); // 5 (спочатку буде вирахувано 1&&0 = 0, потім 5 ||0 = 5)

&&(ТА) можливо використовувати замість if, але краще так не робити:

var x = 1;

(x > 0) && alert( 'Більше' );

!(НЕ) спочатку перетворює аргумент у логічний тип, потім його обертає:

alert( !true ); // false

alert( !0 ); // true

alert( !235 ); // false

if (!(age >= 14 && age <= 90)) = if (age < 14 || age > 90)

Приклади комбінованого застосування операторів:

alert( alert(1) || 2 || alert(3) );

Відповідь: спочатку 1, потім 2.

Рішення: Виклик alert не повертає значення, тобто повертає undefined. Перший оператор ||(АБО) виконає перший alert(1), отримає undefined і піде далі. Оскільки другий операнд 2 є істинним, то розрахунки на ньому закінчаться, результатом undefined || 2 буде 2.

alert( alert(1) && alert(2) );

Відповідь: 1, а потім undefined.

Рішення: Виклик alert не повертає значення, тобто повертає undefined. Тому до правого alert справа не дійде, розрахунки закінчаться на лівому.

alert( null || 2 && 3 || 4 );

Відповідь: 3.

Рішення: Пріоритет оператора && вище, ніж ||, тому він виконається першим. Послідовність розрахунків:

null || 2 && 3 || 4

null || 3 || 4

3

-----

Перетворення типів для примітивів

Перетворення у рядок

false →"false", null → "null", undefined → "undefined" тощо.

Через alert:

var a = true;

alert ( a ); // "true"

Через String():

alert( String(null); // "null"

Через оператор + з (пустим) рядком:

alert( true + "test" ); // "truetest"

alert( 24 + "" ); // "24"

Через метод toString:

alert( n.toString(x)); // x – система числення

Перетворення у число

true → 1, false → 0, null → 0, undefined → NaN.

Для рядків: "" → 0, пробіли з країв та \n (символ переносу рядка) обрізуються, якщо рядок починається із цифри – цифра до першої букви, якщо з букви → NaN.

Через Number():

var a = Number("123"); // 123

Через оператор +:

var a = +"123"; // 123

alert( +true ); // 1

Під час порівняння різних типів:

alert( "\n0 " == 0 ); // true

Використання parseInt та parseFloat – "м'яке" перетворення:

console.log(+"12px") // NaN – букви спричинять помилку

console.log(parseInt("45 px", 10)); // 45 - зайві букви відкидаються, 10 – система числення

console.log( parseInt('FF', 16) ); // 255

console.log(parseFloat("12.45 em")); // 12.45

console.log( parseFloat('12.3.4') ) // 12.3 – помилка на другій крапці

Логічне перетворення

Використання оператору if, який перетворює число 0, порожній рядок, null, undefined та NaN у false, усе інше – в true.

Використання логічних операторів.

Використання подвійного логічного заперечення:

alert( !!"0" ); // true

alert( !!" " ); // true

alert( !!3 ); // true

Використання Boolean():

alert( Boolean(null));

-----

"" + 1 + 0 = "10"

"" - 1 + 0 = -1

true + false = 1

6 / "3" = 2

"2" \* "3" = 6

4 + 5 + "px" = "9px"

"$" + 4 + 5 = "$45"

"4" - 2 = 2

"4px" - 2 = NaN

7 / 0 = Infinity

" -9\n" + 5 = " -9\n5"

" -9\n" - 5 = -14

5 && 2 = 2

2 && 5 = 5

5 || 0 = 5

0 || 5 = 5

null + 1 = 1

undefined + 1 = NaN

null == "\n0\n" = false

+null == +"\n0\n" = true

-----

Цикл for. Пропуск частин

Будь-яку частину циклу можна пропустити, головне – залишити крапку з комою ;

var i = 0;

for (; i < 3; i++) {

alert( i ); // 0, 1, 2

}

for (; i < 3;) {

alert( i );

// цикл перетворився в аналог while (i<3)

}

for (;;) {

// вічний цикл

}

-----

Мітка outer: + команда break outer; дозволяють вийти з усіх вкладених циклів.

outer: for (var i = 0; i < 3; i++) {

for (var j = 0; j < 3; j++) {

var input = prompt('Значение в координатах '+i+','+j, '');

// если отмена ввода или пустая строка -

// завершить оба цикла

if (!input) break outer; // (\*)

}

}

alert('Готово!');

-----

**МЕТОДИ І ВЛАСТИВОСТІ**

Методи – додаткові "функції", доступні через крапку, виклик методу завжди із дужками ().

**ЧИСЛОВІ МЕТОДИ**

Метод **toFixed()** – округлення числа до заданої точності:

var n = 12.345;

alert( n.toFixed(2) ); // "12.35"

alert( n.toFixed(0) ); // "12"

alert( n.toFixed(5) ); // "12.34500"

До методів чисел можна звертатися напряму, але після цілого числа потрібно ставити додаткову крапку або пробіл:

alert( 12.34.toFixed(1) ); // 12.3

alert( 12 .toFixed(1)); // 12.0

alert( 12..toFixed(1) ); // 12.0

alert( 6.35.toFixed(1) ); // 6.3 !!! через безкінечність дробу у двійковій системі

втрачається точність

Метод **isFinite()** - перевірка на не NaN/Infinity/-Infinity:

alert( isFinite(1) ); // true

alert( isFinite(Infinity) ); // false

alert( isFinite(NaN) ); // false

**Перевірка на NaN:**

1. Метод **isNaN()** – перетворює аргумент в число і повертає true, якщо NaN:

var n = 0 / 0;

alert( isNaN(n) ); // true

alert( isNaN("12") ); // false

2. Якщо число не рівне саме собі, то воно NaN:

var n = 0 / 0;

if (n !== n) alert( 'n = NaN!' );

|  |  |
| --- | --- |
| C:\Users\Admin\Desktop\index.png | **Порожній рядок, рядок з пробілів будуть перетворені в 0, тому вважатимуться числами. false, true, null також будуть вважатися числами. Якщо потрібна чітка перевірка на не число, замість isNaN() потрібно використовувати функцію isNumeric():** |

function isNumeric(n) {

return !isNaN(parseFloat(n)) && isFinite(n);

}

Метод **toString()** - переведення з 10-ї, 8-ї, 16-ї систем числення в іншу (від 2 до 36):

alert( 255.toString(16) ); // ff

alert( 4.toString(2) ); // 100

alert( 1234567890.toString(36) ); // kf12oi

**Math.floor** – округлення вниз:

alert( Math.floor(3.1) ); // 3

alert ( Math.floor(-1.2) ); // -2

**Math.ceil** - округлення вверх:

alert( Math.ceil(3.1) ); // 4

**Math.round** – округлення до ближнього цілого:

alert( Math.round(3.1) ); // 3

**Math.acos(x)** - повертає арккосинус x (в радіанах)

**Math.asin(x)** - повертає арксинус x (в радіанах)

**Math.atan(x)** - повертає арктангенс x (в радіанах)

**Math.atan2(y, x)** - повертає кут до крапки (y, x)

**Math.sin(x)** - розраховує синус x

**Math.cos(x)** - розраховує косинус x

**Math.tan(x)** - повертає тангенс x

**Math.sqrt(x)** - повертає квадратний корінь x

**Math.log(x)** - повертає натуральний (по основі e) логарифм x

**Math.pow(x, exp)** - підносить число в степінь, повертає xexp:

Math.pow(2,3) = 8

Math.pow(4, -1/2) = 0.5

**Math.abs(x)** - повертає абсолютне значення числа

**Math.exp(x)** - повертає ex, де e – основа натуральних логарифмів

**Math.max(a, b, c...)** - повертає найбільший зі списку аргументів

**Math.min(a, b, c...)** – повертає найменший зі списку аргументів

**Math.random()** - повертає псевдовипадкове число в інтервалі [0,1) – тобто між 0 (включно) и 1 (не включно). Генератор випадкових чисел ініціалізується поточним часом.

**toLocaleString()** - для красивого виводу чисел в стандарті ECMA 402:

var number = 123456789;

alert( number.toLocaleString() ); // 123 456 789

**МЕТОДИ РЯДКІВ**

методи рядків не видозмінюють рядок, а повертають новий!

Властивість **str.length** – повернення довжини рядка:

alert( "Hello!".length ); // 6

Метод **toUpperCase()** – повернення рядку у верхньому регістрі:

var hello = "Hello!";

alert( hello.toUpperCase() ); // "HELLO!"

Метод **toLowerCase()** – повернення рядку у нижньому регістрі:

alert( "Інтерфейс" [0].toLowerCase() ); // 'і'

**charAt()** – доступ до конкретного символу:

var str = "jQuery";

alert( str.charAt(0) ); // "j"

**str[]** – доступ до конкретного символу:

var str = "Я - браузер!";

alert( str[0] ); // "Я"

**відмінність між charAt() та str[]:**

alert( "".charAt(0) ); // пустий рядок

alert( "" [0] ); // undefined

**indexOf("xхх", n)** – повертає позицію, на якій знаходиться підрядок ххх, або -1, якщо збігів не знайдено, n – не обов'язковий аргумент, який вказує з якої позиції починати шукати. Пошук регістрозалежний.

var str = "Widget with id";

alert( str.indexOf("Widget") ); // 0

alert( str.indexOf("id") ); // 1

alert( str.indexOf("widget") ); // -1

alert(str.indexOf("id", 2)) // 12

**lastindexOf("xхх", n)** – те саме, що і indexOf("xхх", n), тільки проводить пошук з кінця.

**Перевірка на наявність шуканого:**

var str = "Widget";

if (~str.indexOf("get")) { // побітовий оператор не ~n рівний нулю тільки тоді, коли

alert('співпадіння є!' ); // n == -1, а -1 – це умова відсутності збігів. Тому якщо

} // ~n=0, умова if не виконається.

**Пошук усіх співпадінь здійснюється через цикл:**

var str = "Ослик Иа-Иа посмотрел на виадук"; // шукаємо тут

var target = "Иа"; // мета пошуку

var pos = 0;

while (true) {

var foundPos = str.indexOf(target, pos);

if (foundPos == -1) break;

alert( foundPos ); // знайшли на цій позиції

pos = foundPos + 1; // продовжуємо пошук з наступної

}

Метод **substring(start, end)** повертає підрядок з позиції start включно до end не включно. Якщо аргументу end нема – то до кінця:

var str = "stringify";

alert(str.substring(0,1)); // "s", символи з позиції 0 до 1 не включно

alert(str.substring(2)); // "ringify", символи з позиції 2 до кінця

Метод **substr (start, length)** повертає підрядок з позиції start з довжиною у символах length. Якщо аргументу length нема – то до кінця:

var str = "stringify";

str = str.substr(2,4); // ring, з 2-ї позиції 4 символи

alert(str)

Метод **slice(start, end)** повертає підрядок з позиції start включно до end не включно. Якщо аргументу end нема – то до кінця (аналогічно до substring(start, end), але додатково може приймати мінусові значення (щоб виводити значення з кінця рядку)

**Різниця між substring и slice:**

substring(start, end) інтерпретує від'ємні аргументи як рівні нулю, якщо start > end, то аргументи міняються місцями:

alert( "testme".substring(4, -1) ); // "test"

// -1 стає 0 -> substring(4, 0)

// 4 > 0, аргументи міняються місцями -> substring(0, 4) = "test"

slice відраховує від'ємні значення з кінця:

alert( "testme".slice(-2) ); // "me", від 2 позиції з кінця

alert( "testme".slice(1, -1) ); // "estm", від 1 позиції до першої з кінця.

**String.fromCharCode(code)** повертає символ Unicode:

alert( String.fromCharCode(1072) ); // 'а'

**str.charCodeAt(pos)** повертає код Unicode-символу:

alert( "абрикос".charCodeAt(0) ); // 1072, код 'а'

Метод **str.localeCompare()** порівнює рядки не за лексикографічним порівнянням, а за алфавітним. Повертає -1, якщо str1 < str2, 1, якщо str1 > str2 та 0, якщо вони рівні:

var str = "Ёлки";

alert( str.localeCompare("Яблони") ); // -1

**ОБ’ЄКТИ**

1. var о = new Object();
2. var о = {};

var person = {}; // створення об'єкту

person.name = 'Вася'; // додавання пари властивість - значення

person.age = 25; // додавання пари властивість - значення

person['улюблений стиль музики'] = 'Джаз'; // додавання пари властивість - значення

person. улюблений стиль музики = 'Джаз'; // ??? помилка

alert( person.name + ': ' + person.age ); // виведення "Вася: 25"

delete person.age; // видалення пари властивість – значення

Пар "властивість - значення" може бути багато. Один об'єкт може містити інший об'єкт:

var user = {

name: "Jon",

surname: "Smith",

jobs:{

first: "Makler",

second: "Brocker"

}

};

Якщо потрібно **перевірити, чи є в об'єкті певна пара**, використовують оператор **in**:

if ("name" in person) {

alert( "властивість name існує!" );

}

або порівнюють пару з undefined:

var person = {

name: "Василь"

test = undefined;

};

alert( person.lalala === undefined ); // true, такої властивості нема

alert( person.name === undefined ); // false, така властивість є

alert( person.test === undefined ); // true, але test насправді є, тому краще

// користуватися оператором in:

alert( "test" in person); // true

**перебирання елементів об'єкту** - оператор **for in**:

var person = {

firstName: "John",

lastName: "Smith",

age: 25

};

for(var key in person) {

alert(person[key]);

}

**визначення кількості властивостей об'єкта:**

var counter = 0;

for (var key in obj) {

counter++;

}

alert( "Усього властивостей: " + counter );

**перебирання властивостей об'єкта** відбувається у тому порядку, в якому вони прописувалися, але якщо назви властивостей – числа, вони перебиратимуться як числа, за зростанням. Щоб цього уникнути, достатньо перед цифрою поставити знак плюс:

var codes = {

"+7": "Московія",

"+38": "Україна",

"+1": "США"

};

|  |  |
| --- | --- |
| C:\Users\Admin\Desktop\index.png | **Під час копіювання об'єкту копіюються не дані, а посилання на пам'ять, де вони зберігаються:** |

var user = { name: 'Василь' };

var admin = user; // копіюємо об'єкт

admin.name = 'Петрусь'; // змінили дані через admin

alert(user.name); // 'Петрусь', зміни видно в user

Якщо потрібно зробити клон об'єкта:

var user = {

name: "Вася",

age: 30

};

var clone = {}; // новий пустий об'єкт

for (var key in user) { // копіювання властивостей user в clone

clone[key] = user[key];

}

**При "розкритті" властивостей об'єкту в консолі браузер завжди виводить їх поточні (на момент розкриття) значення.** Так відбувається тому, що вивід не робить "копію" поточного вмісту, а лише зберігає посилання на об'єкт.

**МАСИВИ**

var array = []; // створення масиву

var array = new Array("Яблуко", "Груша", "тощо"); // створення масиву

|  |  |
| --- | --- |
| C:\Users\Admin\Desktop\index.png | **Якщо array = new Array(2); має тільки один цифровий аргумент, то замість створення масиву з одним елементом, він створить масив без елементів з довжиною, вказаною в аргументі.** |

var array = ['one', 'two', 'three']; // нумерація елементів починається з нуля

var array = ['one', 2, 3, true]; // масиви можуть мати змішані типи даних

alert(array); // виведення значення всього масиву

alert(array[0]); // виведення значення першого елементу масиву

array[0] = 'trololo'; // переприсвоєння значення елементу масиву

delete arr[1]; // значення елементу з індексом 1 видалено, утворюється дірка (undefined)

array.indexOf(value); // виведення індексу елементу

|  |  |
| --- | --- |
| C:\Users\Admin\Desktop\index.png | **Довжина length – не кількість елементів масиву, а останній індекс + 1** |

alert(array.length); // визначення довжини масиву

array.push('four', 'five'); // додавання елементу (ів) у кінець масиву

array.pop(); // видалення елементу з кінця масиву

array.unshift('zero'); // додавання елементу (ів) на початок масиву

array.shift(); // видалення елементу з початку масиву

в один і той же масив можна вносити різні властивості:

array [99999] = 5; // властивість з номерним іменем 99999

array.age = 25; // властивість з рядковим іменем age

властивості можна задавати не по порядку:

var a = [];

a[0] = 0;

a[5] = 5;

alert( a ); // 0,,,,,5

перебирання елементів масиву:

var arr = ["Яблуко", "Апельсин", "Груша"];

for (var i = 0; i < arr.length; i++) {

alert( arr[i] );

}

скорочення довжини масиву:

var arr = [1, 2, 3, 4, 5];

arr.length = 2; // обрізати до 2 елементів

alert( arr ); // [1, 2]

багатовимірні масиви:

var matrix = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9]

];

alert( matrix[1][1] ); // 5

Метод **split()** – розділення рядку на масив із розділювальним символом. Другий аргумент (не обов’язковий) вказує, після якого символу припинити сепарацію:

alert( "a,b,c,d".split(',', 2) ); // a,b

**Розбиття рядка на окремі букви** – split з пустим рядком:

var str = "тест";

alert( str.split('') ); // т,е,с,т

Метод **join()** – склеювання масиву у рядок. (';') – вказується символ, яким буде розділено елементи. Якщо () – як розділовий знак використовується кома, якщо ('') – склеювання без знаків і пробілів:

var arr = ['Маша', 'Петя', 'Марина', 'Василий'];

var str = arr.join(';');

alert( str ); // Маша;Петя;Марина;Василий

**new Array + join = повторення рядку**

alert( new Array(4).join("ля") ); // ляляля, new Array(4) робить масив без

елементів довжини 4, який join об’єднує в рядок, вставляючи між його елементами рядок ля

Метод **splice** – видалення, заміна, вставка елементів:

array.splice(x, y); // видалення з позиції x y елементів

array.splice(x,y, "elem-1", …, "elem-n"); // видалення з позиції x y елементів, замість

них в тіло масиву вставити елементи "elem-1", …, "elem-n". Якщо y=0, видалення не буде, буде тільки заміна. Параметр х може бути мінусовим, тоді він відраховує від кінця масиву

Метод **slice(begin, end) –** копіювання частини масиву від begin включно до end не включно, якщо не вказати end – то до кінця. Параметр **begin** може бути мінусовим, тоді він відраховує від кінця масиву:

var arr = ("Яблуко", "Груша", "Слива", "Абрикос");

var arr2 = arr.slice(1, 3);

alert( arr2 ); // Груша, Слива

array.sort(); // розташувати елементи масиву по порядку

(сприйняття елементів як рядків)

array.sort(function (a, b){ // розташувати елементи масиву по порядку

return a-b; (сприйняття елементів як чисел): функція

}); порівнює сусідні значення, якщо різниця додатня,

елементи міняються місцями

var arr = [1, 2, 3, 4, 5]; // випадкове сортування масиву

function compareRandom(a, b) {

return Math.random() - 0.5;

}

arr.sort(compareRandom);

alert( arr );

array.reverse(); // розташувати масив у зворотному порядку

Метод **arr.concat(value1, value2, … valueN)** створює новий масив, додаючи до старого нові значення, якщо серед значень є вкладені масиви, він додає їх як елементи:

var arr = [1, 2];

var newArr = arr.concat([3, 4], 5); // те саме, що arr.concat(3,4,5)

alert( newArr ); // 1,2,3,4,5

Метод **arr.indexOf(searchElement[, fromIndex])** повертає номер елемента searchElement в масиві arr або -1, якщо його нема. Пошук починається з номеру fromIndex, якщо він вказаний. Якщо ні – з початку масиву. Для пошуку використовується строге порівняння ===.

var arr = [1, 0, false];

alert( arr.indexOf(0) ); // 1

alert( arr.indexOf(false) ); // 2

alert( arr.indexOf(null) ); // -1

Метод **arr.lastindexOf(searchElement[, fromIndex])** робить те саме, що і **arr.indexOf(searchElement[, fromIndex])**, але справа на ліво.

Метод **Object.keys(obj)** вертає властивості об'єкта як елементи масиву:

var user = {

name: "Петя",

age: 30

}

var keys = Object.keys(user);

alert( keys ); // name, age – масив

**ПЕРЕБИРАЮЧІ МЕТОДИ МАСИВІВ**

Метод **arr.forEach(callback (item, i, arr){})** – **перебирання масиву.**

Метод нічого не повертає. Для кожного елементу масиву викликає функціию callback, якій передає три параметри: item – черговий елемент масиву, i – його номер, arr – масив, який перебирається:

var arr = ["Яблуко", "Апельсин", "Груша"];

arr.forEach(function(item, i, arr) {

alert( i + ": " + item + " (масив:" + arr + ")" );

});

Метод **arr.filter(callback (){}**) **– фільтрація масиву через функцію.**

Створює новий масив, в який увійдуть тільки ті елементи arr, для яких виклик callback()поверне true.

var arr = [1, -1, 2, -2, 3];

var positiveArr = arr.filter(function(number) {

return number > 0;

});

alert( positiveArr ); // 1,2,3

Метод **arr.map(callback(){}**) - **трансформація** **масиву.**

Створює новий масив, який буде складатися з результатів виклику callback() для кожного елемента arr.

var names = ['HTML', 'CSS', 'JavaScript'];

var nameLengths = names.map(function(name) {

return name.length;

});

отримали масив з довжинами

alert( nameLengths ); // 4,3,10

Метод **arr.every(callback(){})** **повертає true**, якщо виклик callback поверне true для кожного елементу arr.

Метод **arr.some(callback(){})** **повертає true**, якщо виклик callback поверне true для якого-небудь елемента arr.

var arr = [1, -1, 2, -2, 3];

function isPositive(number) {

return number > 0;

}

alert( arr.every(isPositive) ); // false, не всі більше нуля

alert( arr.some(isPositive) ); // true, є мінімум одне більше нуля

Метод **arr.reduce(callback(){}**) – **послідовна обробка кожного елементу масиву зі збереженням проміжного результату.**

Аргументи функції callback(previousValue, currentItem, index, arr):

previousValue – останній результат виклику функції, він же "проміжний результат"

currentItem – поточний елемент масиву

index – номер поточного елементу

arr – оброблюваний масив

Крім callback методу можна передати "початкове значення" – аргумент initialValue. Якщо він є, то на першому виклику значення previousValue буде рівне initialValue, а якщо у reduce нема другого аргументу, то воно рівне першому елементу масиву, а перебирання починається з другого.

var arr = [1, 2, 3, 4, 5]

var result = arr.reduce(function(sum, current) {

return sum + current;

}, 0); // 0 – другий аргумент reduce (початкове sum)

alert( result ); // 15

**Псевдомасив аргументів "arguments"**

function sayHi() {

for (var i = 0; i < arguments.length; i++) {

alert( "Привет, " + arguments[i] );

}

}

sayHi("Винни", "Пятачок"); // 'Привет, Винни', 'Привет, Пятачок'

SORAX

ПРИМІТИВИ

|  |  |
| --- | --- |
| console.log(n.toExponential(4)); | перетворення на експоненційне число |
| console.log(string.replace("xxx", "yyy")); | заміна в рядку xxx на yyy |

## Язык JavaScript

* [Введение](http://learn.javascript.ru/getting-started)
  + [Введение в JavaScript](http://learn.javascript.ru/intro)
  + [Справочники и спецификации](http://learn.javascript.ru/pre-coding)
  + [Редакторы для кода](http://learn.javascript.ru/editor)
  + [Консоль разработчика](http://learn.javascript.ru/devtools)
* [Основы JavaScript](http://learn.javascript.ru/first-steps)
  + [Привет, мир!](http://learn.javascript.ru/hello-world)
  + [Внешние скрипты, порядок исполнения](http://learn.javascript.ru/external-script)
  + [Структура кода](http://learn.javascript.ru/structure)
  + [Современный стандарт, "use strict"](http://learn.javascript.ru/strict-mode)
  + [Переменные](http://learn.javascript.ru/variables)
  + [Правильный выбор имени переменной](http://learn.javascript.ru/variable-names)
  + [Шесть типов данных, typeof](http://learn.javascript.ru/types-intro)
  + [Основные операторы](http://learn.javascript.ru/operators)
  + [Операторы сравнения и логические значения](http://learn.javascript.ru/comparison)
  + [Побитовые операторы](http://learn.javascript.ru/bitwise-operators)
  + [Взаимодействие с пользователем: alert, prompt, confirm](http://learn.javascript.ru/uibasic)
  + [Условные операторы: if, '?'](http://learn.javascript.ru/ifelse)
  + [Логические операторы](http://learn.javascript.ru/logical-ops)
  + [Преобразование типов для примитивов](http://learn.javascript.ru/types-conversion)
  + [Циклы while, for](http://learn.javascript.ru/while-for)
  + [Конструкция switch](http://learn.javascript.ru/switch)
  + [Функции](http://learn.javascript.ru/function-basics)
  + [Функциональные выражения](http://learn.javascript.ru/function-declaration-expression)
  + [Рекурсия, стек](http://learn.javascript.ru/recursion)
  + [Именованные функциональные выражения](http://learn.javascript.ru/named-function-expression)
  + [Всё вместе: особенности JavaScript](http://learn.javascript.ru/javascript-specials)
* [Качество кода](http://learn.javascript.ru/writing-js)
  + [Отладка в браузере Chrome](http://learn.javascript.ru/debugging-chrome)
  + [Советы по стилю кода](http://learn.javascript.ru/coding-style)
  + [Как писать неподдерживаемый код?](http://learn.javascript.ru/write-unmain-code)
  + [Автоматические тесты при помощи chai и mocha](http://learn.javascript.ru/testing)
* [Структуры данных](http://learn.javascript.ru/data-structures)
  + [Введение в методы и свойства](http://learn.javascript.ru/properties-and-methods)
  + [Числа](http://learn.javascript.ru/number)
  + [Строки](http://learn.javascript.ru/string)
  + [Объекты как ассоциативные массивы](http://learn.javascript.ru/object)
  + [Объекты: перебор свойств](http://learn.javascript.ru/object-for-in)
  + [Объекты: передача по ссылке](http://learn.javascript.ru/object-reference)
  + [Массивы с числовыми индексами](http://learn.javascript.ru/array)
  + [Массивы: методы](http://learn.javascript.ru/array-methods)
  + [Массив: перебирающие методы](http://learn.javascript.ru/array-iteration)
  + [Псевдомассив аргументов "arguments"](http://learn.javascript.ru/arguments-pseudoarray)
  + [Дата и Время](http://learn.javascript.ru/datetime)
* [Замыкания, область видимости](http://learn.javascript.ru/functions-closures)
  + [Глобальный объект](http://learn.javascript.ru/global-object)
  + [Замыкания, функции изнутри](http://learn.javascript.ru/closures)
  + [[[Scope]] для new Function](http://learn.javascript.ru/scope-new-function)
  + [Локальные переменные для объекта](http://learn.javascript.ru/closures-usage)
  + [Модули через замыкания](http://learn.javascript.ru/closures-module)
  + [Управление памятью в JavaScript](http://learn.javascript.ru/memory-management)
  + [Устаревшая конструкция "with"](http://learn.javascript.ru/with)
* [Методы объектов и контекст вызова](http://learn.javascript.ru/objects-more)
  + [Методы объектов, this](http://learn.javascript.ru/object-methods)
  + [Преобразование объектов: toString и valueOf](http://learn.javascript.ru/object-conversion)
  + [Создание объектов через "new"](http://learn.javascript.ru/constructor-new)
  + [Дескрипторы, геттеры и сеттеры свойств](http://learn.javascript.ru/descriptors-getters-setters)
  + [Статические и фабричные методы](http://learn.javascript.ru/static-properties-and-methods)
  + [Явное указание this: "call", "apply"](http://learn.javascript.ru/call-apply)
  + [Привязка контекста и карринг: "bind"](http://learn.javascript.ru/bind)
  + [Функции-обёртки, декораторы](http://learn.javascript.ru/decorators)
* [Некоторые другие возможности](http://learn.javascript.ru/js-misc)
  + [Типы данных: [[Class]], instanceof и утки](http://learn.javascript.ru/class-instanceof)
  + [Формат JSON, метод toJSON](http://learn.javascript.ru/json)
  + [setTimeout и setInterval](http://learn.javascript.ru/settimeout-setinterval)
  + [Запуск кода из строки: eval](http://learn.javascript.ru/eval)
  + [Перехват ошибок, "try..catch"](http://learn.javascript.ru/exception)
* [ООП в функциональном стиле](http://learn.javascript.ru/oop)
  + [Введение](http://learn.javascript.ru/about-oop)
  + [Внутренний и внешний интерфейс](http://learn.javascript.ru/internal-external-interface)
  + [Геттеры и сеттеры](http://learn.javascript.ru/getters-setters)
  + [Функциональное наследование](http://learn.javascript.ru/functional-inheritance)
* [ООП в прототипном стиле](http://learn.javascript.ru/prototypes)
  + [Прототип объекта](http://learn.javascript.ru/prototype)
  + [Свойство F.prototype и создание объектов через new](http://learn.javascript.ru/new-prototype)
  + [Встроенные "классы" в JavaScript](http://learn.javascript.ru/native-prototypes)
  + [Свои классы на прототипах](http://learn.javascript.ru/classes)
  + [Наследование классов в JavaScript](http://learn.javascript.ru/class-inheritance)
  + [Проверка класса: "instanceof"](http://learn.javascript.ru/instanceof)
  + [Свои ошибки, наследование от Error](http://learn.javascript.ru/oop-errors)
  + [Примеси](http://learn.javascript.ru/mixins)
* [Современные возможности ES-2015](http://learn.javascript.ru/es-modern)
  + [ES-2015 сейчас](http://learn.javascript.ru/es-modern-usage)
  + [Переменные: let и const](http://learn.javascript.ru/let-const)
  + [Деструктуризация](http://learn.javascript.ru/destructuring)
  + [Функции](http://learn.javascript.ru/es-function)
  + [Строки](http://learn.javascript.ru/es-string)
  + [Объекты и прототипы](http://learn.javascript.ru/es-object)
  + [Классы](http://learn.javascript.ru/es-class)
  + [Тип данных Symbol](http://learn.javascript.ru/symbol)
  + [Итераторы](http://learn.javascript.ru/iterator)
  + [Set, Map, WeakSet и WeakMap](http://learn.javascript.ru/set-map)
  + [Promise](http://learn.javascript.ru/promise)
  + [Генераторы](http://learn.javascript.ru/generator)
  + [Модули](http://learn.javascript.ru/modules)
  + [Proxy](http://learn.javascript.ru/proxy)

## Документ, события, интерфейсы

* [Документ и объекты страницы](http://learn.javascript.ru/document)
  + [Окружение: DOM, BOM и JS](http://learn.javascript.ru/browser-environment)
  + [Дерево DOM](http://learn.javascript.ru/dom-nodes)
  + [Работа с DOM из консоли](http://learn.javascript.ru/dom-console)
  + [Навигация по DOM-элементам](http://learn.javascript.ru/traversing-dom)
  + [Поиск: getElement\* и querySelector\* и не только](http://learn.javascript.ru/searching-elements-dom)
  + [Внутреннее устройство поисковых методов](http://learn.javascript.ru/searching-elements-internals)
  + [Свойства узлов: тип, тег и содержимое](http://learn.javascript.ru/basic-dom-node-properties)
  + [Современный DOM: полифиллы](http://learn.javascript.ru/dom-polyfill)
  + [Атрибуты и DOM-свойства](http://learn.javascript.ru/attributes-and-custom-properties)
  + [Методы contains и compareDocumentPosition](http://learn.javascript.ru/compare-document-position)
  + [Добавление и удаление узлов](http://learn.javascript.ru/modifying-document)
  + [Мультивставка: insertAdjacentHTML и DocumentFragment](http://learn.javascript.ru/multi-insert)
  + [Метод document.write](http://learn.javascript.ru/document-write)
  + [Стили, getComputedStyle](http://learn.javascript.ru/styles-and-classes)
  + [Размеры и прокрутка элементов](http://learn.javascript.ru/metrics)
  + [Размеры и прокрутка страницы](http://learn.javascript.ru/metrics-window)
  + [Координаты в окне](http://learn.javascript.ru/coordinates)
  + [Координаты в документе](http://learn.javascript.ru/coordinates-document)
  + [Итого](http://learn.javascript.ru/dom-cheatsheet)
* [Основы работы с событиями](http://learn.javascript.ru/events-and-interfaces)
  + [Введение в браузерные события](http://learn.javascript.ru/introduction-browser-events)
  + [Порядок обработки событий](http://learn.javascript.ru/events-and-timing-depth)
  + [Объект события](http://learn.javascript.ru/obtaining-event-object)
  + [Всплытие и перехват](http://learn.javascript.ru/event-bubbling)
  + [Делегирование событий](http://learn.javascript.ru/event-delegation)
  + [Приём проектирования "поведение"](http://learn.javascript.ru/behavior)
  + [Действия браузера по умолчанию](http://learn.javascript.ru/default-browser-action)
  + [Генерация событий на элементах](http://learn.javascript.ru/dispatch-events)
* [События в деталях](http://learn.javascript.ru/event-details)
  + [Мышь: клики, кнопка, координаты](http://learn.javascript.ru/mouse-clicks)
  + [Мышь: отмена выделения, невыделяемые элементы](http://learn.javascript.ru/unselectable)
  + [Мышь: движение mouseover/out, mouseenter/leave](http://learn.javascript.ru/mousemove-mouseover-mouseout-mouseenter-mouseleave)
  + [Мышь: Drag'n'Drop](http://learn.javascript.ru/drag-and-drop)
  + [Мышь: Drag'n'Drop более глубоко](http://learn.javascript.ru/drag-and-drop-objects)
  + [Мышь: колёсико, событие wheel](http://learn.javascript.ru/mousewheel)
  + [Мышь: IE8-, исправление события](http://learn.javascript.ru/fixevent)
  + [Прокрутка: событие scroll](http://learn.javascript.ru/onscroll)
  + [Клавиатура: keyup, keydown, keypress](http://learn.javascript.ru/keyboard-events)
  + [Загрузка документа: DOMContentLoaded, load, beforeunload, unload](http://learn.javascript.ru/onload-ondomcontentloaded)
  + [Загрузка скриптов, картинок, фреймов: onload и onerror](http://learn.javascript.ru/onload-onerror)
* [Формы, элементы управления](http://learn.javascript.ru/forms-controls)
  + [Навигация и свойства элементов формы](http://learn.javascript.ru/form-elements)
  + [Фокусировка: focus/blur](http://learn.javascript.ru/focus-blur)
  + [Изменение: change, input, cut, copy, paste](http://learn.javascript.ru/events-change)
  + [Формы: отправка, событие и метод submit](http://learn.javascript.ru/forms-submit)
* [Создание графических компонентов](http://learn.javascript.ru/widgets)
  + [Введение](http://learn.javascript.ru/architect-intro)
  + [Графические компоненты](http://learn.javascript.ru/widgets-structure)
  + [Вёрстка графических компонентов](http://learn.javascript.ru/widgets-markup)
  + [Шаблонизатор LoDash](http://learn.javascript.ru/template-lodash)
  + [Коллбэки и события на компонентах](http://learn.javascript.ru/custom-events)
  + [Что изучать дальше](http://learn.javascript.ru/what-next)

## Дополнительно

* [Веб-компоненты: взгляд в будущее](http://learn.javascript.ru/webcomponents)
  + [С высоты орбитального полёта](http://learn.javascript.ru/webcomponents-intro)
  + [Пользовательские элементы: Custom Elements](http://learn.javascript.ru/webcomponent-core)
  + [Shadow DOM](http://learn.javascript.ru/shadow-dom)
  + [Шаблоны <template>](http://learn.javascript.ru/template-tag)
  + [Стили и селекторы](http://learn.javascript.ru/css-scoping)
  + [Импорты](http://learn.javascript.ru/link-import)
  + [Веб-компонент в сборе](http://learn.javascript.ru/webcomponent-build)
* [AJAX и COMET](http://learn.javascript.ru/ajax)
  + [Введение в AJAX и COMET](http://learn.javascript.ru/ajax-intro)
  + [Node.JS для решения задач](http://learn.javascript.ru/ajax-nodejs)
  + [Основы XMLHttpRequest](http://learn.javascript.ru/ajax-xmlhttprequest)
  + [XMLHttpRequest POST, формы и кодировка](http://learn.javascript.ru/xhr-forms)
  + [XMLHttpRequest: кросс-доменные запросы](http://learn.javascript.ru/xhr-crossdomain)
  + [XMLHttpRequest: индикация прогресса](http://learn.javascript.ru/xhr-onprogress)
  + [XMLHttpRequest: возобновляемая закачка](http://learn.javascript.ru/xhr-resume)
  + [COMET с XMLHttpRequest: длинные опросы](http://learn.javascript.ru/xhr-longpoll)
  + [WebSocket](http://learn.javascript.ru/websockets)
  + [Протокол JSONP](http://learn.javascript.ru/ajax-jsonp)
  + [Server Side Events -- события с сервера](http://learn.javascript.ru/server-sent-events)
  + [IFRAME для AJAX и COMET](http://learn.javascript.ru/ajax-iframe)
  + [Атака CSRF](http://learn.javascript.ru/csrf)
  + [Метод fetch: замена XMLHttpRequest](http://learn.javascript.ru/fetch)
  + [Таблица транспортов и их возможностей](http://learn.javascript.ru/ajax-summary)
* [Анимация](http://learn.javascript.ru/animation)
  + [Кривые Безье](http://learn.javascript.ru/bezier)
  + [CSS-анимации](http://learn.javascript.ru/css-transitions)
  + [JS-Анимация](http://learn.javascript.ru/js-animation)
* [Оптимизация](http://learn.javascript.ru/optimize)
  + [Введение](http://learn.javascript.ru/optimize-intro)
  + [Как работают сжиматели JavaScript](http://learn.javascript.ru/minification)
  + [Улучшаем сжатие кода](http://learn.javascript.ru/better-minification)
  + [Утечки памяти](http://learn.javascript.ru/memory-leaks)
  + [Утечки памяти при использовании jQuery](http://learn.javascript.ru/memory-leaks-jquery)
  + [Очистка памяти при removeChild/innerHTML](http://learn.javascript.ru/memory-removechild-innerhtml)
  + [GCC: продвинутые оптимизации](http://learn.javascript.ru/gcc-advanced-optimization)
  + [GCC: статическая проверка типов](http://learn.javascript.ru/gcc-check-types)
  + [GCC: интеграция с Google Closure Library](http://learn.javascript.ru/gcc-closure-library)
* [Окна и Фреймы](http://learn.javascript.ru/frames-and-windows)
  + [Открытие окон и методы window](http://learn.javascript.ru/window-methods)
  + [Общение между окнами и фреймами](http://learn.javascript.ru/iframes)
  + [Кросс-доменные ограничения и их обход](http://learn.javascript.ru/same-origin-policy)
  + [Общение окон с разных доменов: postMessage](http://learn.javascript.ru/cross-window-messaging-with-postmessage)
  + [Привлечение внимания к окну](http://learn.javascript.ru/window-focus)
  + [Атака Clickjacking и защита от неё](http://learn.javascript.ru/clickjacking)
* [CSS для JavaScript-разработчика](http://learn.javascript.ru/css-for-js)
  + [О чём пойдёт речь](http://learn.javascript.ru/css-why)
  + [Единицы измерения: "px", "em", "rem" и другие](http://learn.javascript.ru/css-units)
  + [Все значения свойства "display"](http://learn.javascript.ru/display)
  + [Свойство "float"](http://learn.javascript.ru/float)
  + [Свойство "position"](http://learn.javascript.ru/position)
  + [Центрирование горизонтальное и вертикальное](http://learn.javascript.ru/css-center)
  + [Свойства "font-size" и "line-height"](http://learn.javascript.ru/font-size-line-height)
  + [Свойство white-space](http://learn.javascript.ru/white-space)
  + [Свойство "outline"](http://learn.javascript.ru/outline)
  + [Свойство "box-sizing"](http://learn.javascript.ru/box-sizing)
  + [Свойство "margin"](http://learn.javascript.ru/margin)
  + [Лишнее место под IMG](http://learn.javascript.ru/space-under-img)
  + [Свойство "overflow"](http://learn.javascript.ru/overflow)
  + [Особенности свойства "height" в %](http://learn.javascript.ru/height-percent)
  + [Знаете ли вы селекторы?](http://learn.javascript.ru/css-selectors)
  + [CSS-спрайты](http://learn.javascript.ru/css-sprite)
  + [Правила форматирования CSS](http://learn.javascript.ru/css-format)
* [Сундучок с инструментами](http://learn.javascript.ru/tools)
  + [Полезные расширения Firefox и Chrome](http://learn.javascript.ru/tools-browser-extensions)
  + [Скриптуемый отладочный прокси Fiddler](http://learn.javascript.ru/fiddler)
  + [IE HTTP Analyzer](http://learn.javascript.ru/ie-http-analyzer)
* [Регулярные выражения](http://learn.javascript.ru/regular-expressions-javascript)
  + [Паттерны и флаги](http://learn.javascript.ru/regexp-introduction)
  + [Методы RegExp и String](http://learn.javascript.ru/regexp-methods)
  + [Классы и спецсимволы](http://learn.javascript.ru/regexp-character-classes)
  + [Наборы и диапазоны [...]](http://learn.javascript.ru/regexp-character-sets-and-ranges)
  + [Квантификаторы +, \*, ? и {n}](http://learn.javascript.ru/regexp-quantifiers)
  + [Жадные и ленивые квантификаторы](http://learn.javascript.ru/regexp-greedy-and-lazy)
  + [Скобочные группы](http://learn.javascript.ru/regexp-groups)
  + [Обратные ссылки: \n и $n](http://learn.javascript.ru/regexp-backreferences)
  + [Альтернация (или) |](http://learn.javascript.ru/regexp-alternation)
  + [Начало строки ^ и конец $](http://learn.javascript.ru/regexp-ahchors)
  + [Многострочный режим, флаг "m"](http://learn.javascript.ru/regexp-multiline-mode)
  + [Предпросмотр (неготово)](http://learn.javascript.ru/regexp-lookahead)
  + [Чёрная дыра бэктрекинга](http://learn.javascript.ru/regexp-infinite-backtracking-problem)
* [О всякой всячине](http://learn.javascript.ru/extra)
  + [Эволюция шаблонных систем для JavaScript](http://learn.javascript.ru/templates)
  + [Книги по JS, HTML/CSS и не только](http://learn.javascript.ru/books)
  + [Асинхронное выполнение: setImmediate](http://learn.javascript.ru/setimmediate)
  + [Позднее связывание "bindLate"](http://learn.javascript.ru/bind-late)
  + [Sublime Text: шпаргалка](http://learn.javascript.ru/sublime)
  + [Выделение: Range, TextRange и Selection](http://learn.javascript.ru/range-textrange-selection)
  + [Применяем ООП: Drag'n'Drop++](http://learn.javascript.ru/drag-and-drop-plus)
  + [Куки, document.cookie](http://learn.javascript.ru/cookie)
  + [Intl: интернационализация в JavaScript](http://learn.javascript.ru/intl)
  + [Особенности регулярных выражений в Javascript](http://learn.javascript.ru/regexp-specials)

|  |  |
| --- | --- |
| СЕЛЕКТОРИ |  |
| \* { } | універсальний селектор - обирає усі елементи сторінки |
| #id\_name { } | звертання до id="id\_name". У межах одного html-файлу усі id мають унікальні імена |
| div#container { } | звертання до <div id="container"> |
| .class\_name { } | звертання до елементів класу class="class\_name". Class присвоюється кільком елементам, яким потрібно задати однакові параметри |
| div.class\_name { } | звертання до усіх div-елементів класу class="class\_name". |
| p { } | звертання до усіх елементів <p> |
| input[type="text"]{ } | звертання до усіх елементів <input> з типом type="text" |
| a[href="https://google.com"] { } | звертання до конкретного посилання |
|  |  |
| ВАЖЛИВІСТЬ СЕЛЕКТОРІВ |  |
| !important  inline-style  id  class / pseudoclass / attribute  tag  browser default |  |
|  |  |
| РЕГУЛЯРНІ ВИРАЗИ |  |
| a[href\*="trololo"] { } | звертання до усіх посилань, які мають в своєму тілі "trololo" |
| [class\*="col-"] | \* - звертання до усіх класів, які мають в своїй назві "col-". Шуканий набір символів може бути в будь-якому місці, бути окремим словом, чи частиною. |
| a[href^="https"]{ } | ^ - звертання до усіх посилань, які мають на початку тіла "https". Шукане слово може бути відділене дефісом, пробілом, або бути частиною більшого слова. |
| a[title|="my-title"] { } | | - звертання до усіх посилань, у яких значення атрибуту title буде починатися з my-title, слово типу my-title-first теж підійде, але my-titlefirst уже ні. |
| img[src$=".gif"]{ } | $ - звертання до усіх зображень, які мають розширення .gif. Шуканий набір символів не повинен бути окремим словом. |
| a[data-filetype="image"] { } | звертання до усіх посилань <a href="image.jpg" data-filetype="image"> картинка </a>. Для виділення посилань на усі картинки у самі посилання вставляємо data-filetype="image" |
| a[data-info~="external"] { } | ~ - звертання до певного значення атрибуту зі списку значень, напр. <a href="image.jpg" data-info="external image"> Click Me </a>/  Слово повинно бути відокремлене пробілами, слово типу external-trololo не підійде. |
|  |  |
| СЕЛЕКТОРИ-КОМБІНАТОРИ |  |
| div a { } | звертання до усіх елементів <a>, які знаходяться в елементах <div>, причому будь-якого рівня вкладеності |
| div > a { } | звертання до усіх елементів <a>, які є дочірніми щодо елементів <div> |
| div > a > span { } | звертання до усіх елементів <span>, які є дочірніми щодо елементів <a>, які у свою чергу є є дочірніми щодо елементів <div> |
| tag1 ~ tag2 | звертання до усіх елементів <tag2>, які є братніми до <tag> і йдуть після нього |
| tag1 + tag2 | звертання до одного елементу <tag2>, який є братнім до <tag> і йде після нього |
|  |  |
| ПСЕВДОЕЛЕМЕНТИ |  |
| div::after { } | створення неіснуючого елементу в контейнері div та поміщення його в кінець |
| div::before { } | створення неіснуючого елементу в контейнері div та поміщення його на початок |
| p::first-line { } | звернення до першого рядку, може змінити font properties, color properties, background properties, word-spacing, letter-spacing, text-decoration, vertical-align, text-transform, line-height, clear |
| p::first-letter { } | звернення до першої букви, може змінити font properties, color properties, background properties, margin properties, padding properties, border properties, text-decoration, vertical-align (only if "float" is "none"), text-transform, line-height, float, clear |
| input::placeholder { } | звернення до тексту-підказки у формі |
| p::selection { } | звернення до тексту, виділеного мишкою, може змінити color, background, cursor, outline |
|  |  |
| СЕЛЕКТОРИ ПСЕВДОКЛАСІВ  (user interface) |  |
| a:link { } | звертання до невідвіданих посилань |
| a:visted { } | звертання до відвіданих посилань |
| a:hover { } | звертання до посилань, на яких зафіксована мишка. :hover застосовується до усіх об'єктів |
| a:active { } | звертання до посилань, на яких мишка натиснена і не відпущена. :active застосовується до усіх об'єктів |
| input[type=radio]:checked | звернення до radio або checkbox, причому тільки вибраних |
| div:not(#container){ } | псевдоклас заперечення: вибір усіх div за винятком <div id="container">. not() не дозволяє вписувати у себе селектори-комбінатори |
| \*:not(p) { } | псевдоклас заперечення: вибір усіх ел-тів, крім параграфів |
| li:not(:last-child) { } | звернення до усіх елементів <li> крім останього |
| span:empty { } | звернення тільки до пустих елементів: <span></span> |
| h2:target { } | звернення до виділеного елементу:  <a href="#h1">заголовок</a>  <h2 id="h1">заголовок</h2>  при натисненні на посилання, браузер перейде до заголовку і застосує до нього потрібні стилі |
| input:focus { } | звернення до елементу форми, в якому знаходиться курсор |
| input:enabled { } | звернення до усіх незаблокованих полів форм (в яких нема атрибуту disabled) |
| textarea:disabled { } | звернення до усіх заблокованих полів форм |
| input:invalid { } | звернення до полів форми, заповнених не правильно |
| input:valid { } | звернення до полів форми, заповнених правильно |
| input:read-write {} | звернення до полів форми, які можливо змінювати |
| input:read-only {} | звернення до полів форми, у яких є атрибут readonly |
| input:required { } | звернення до полів форми, заповнення яких є обов'язковим (атрибут required) |
| input:optional { } | звернення до полів форми, заповнення яких не є обов'язковим |
| input:out-of-range { } | звернення до полів форми, де користувач вийшов за межі min або max:  input:out-of-range + label::after {  content: 'невірне значення!'; /\* виводимо повідомлення \*/  color: #ec008c; /\* колір повідомлення \*/  margin-left: 0.5em; /\* відступ зліва \*/  }\*/ |
| input:in-range { } | звернення до полів форми, де користувач залишився у межах min або max |
|  |  |
| СЕЛЕКТОРИ ПСЕВДОКЛАСІВ  (структурні)  загальна формула :xxx(an+b) {} |  |
| li:nth-child(3) { } | звертання до третього з початку елементу <li> |
| li:nth-last-child(3) { } | звертання до третього з кінця елементу <li> |
| li:nth-child(2n) { } | звертання до усіх парних елементів <li> |
| li:nth-child(even) | звертання до усіх парних елементів <li> |
| li:nth-child(2n+1) { } | звертання до усіх непарних елементів <li> |
| li:nth-child(odd) { } | звертання до усіх непарних елементів <li> |
| li:nth-child(3n) { } | звертання до кожного третього елементу <li> |
| li:nth-child(-n+10) { } | звертання до елементів <li> з 1 по 10 |
|  |  |
| ul:nth-of-type(3) { } | звертання до третього елементу <ul> |
|  |  |
| !!! до -of-type можна засосовувати усі ті прийоми, що і до -child.  !!! -last- змінює напрям відліку, усі вищевказані псевдокласи можна модифікувати під нього. | |
|  |  |
| li:first-child { } = li:nth-child(1) { }  li:last-child { } = li:nth-last-child(1) { } | |
|  |  |
| div p:only-child { } | звертання до усіх <p>, які у своїх <div> є єдиними дочірніми |
| div p:only-of-type { } | звертання до усіх <p>, які у своїх <div> є єдиними дочірніми <p> (інші типи елементів не враховуються) |
|  |  |
| різниця між -child та -of-type:  -child 1. спочатку проводить селектування по (an+b) 2. потім, якщо тип збігається, застосовує до нього стиль.  -of-type 1. спочатку вибирає усі елементи одного типу 2. потім проводить селектування по (an+b). | |
|  |  |