[**lodash**](https://lodash.com/)**v3.10.1**

**Array**

* [\_.chunk](https://github.com/lodash/lodash/tree/3.10.1/doc#_chunkarray-size1)
* [\_.compact](https://github.com/lodash/lodash/tree/3.10.1/doc#_compactarray)
* [\_.difference](https://github.com/lodash/lodash/tree/3.10.1/doc#_differencearray-values)
* [\_.drop](https://github.com/lodash/lodash/tree/3.10.1/doc#_droparray-n1)
* [\_.dropRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_droprightarray-n1)
* [\_.dropRightWhile](https://github.com/lodash/lodash/tree/3.10.1/doc#_droprightwhilearray-predicate_identity-thisarg)
* [\_.dropWhile](https://github.com/lodash/lodash/tree/3.10.1/doc#_dropwhilearray-predicate_identity-thisarg)
* [\_.fill](https://github.com/lodash/lodash/tree/3.10.1/doc#_fillarray-value-start0-endarraylength)
* [\_.findIndex](https://github.com/lodash/lodash/tree/3.10.1/doc#_findindexarray-predicate_identity-thisarg)
* [\_.findLastIndex](https://github.com/lodash/lodash/tree/3.10.1/doc#_findlastindexarray-predicate_identity-thisarg)
* [\_.first](https://github.com/lodash/lodash/tree/3.10.1/doc#_firstarray)
* [\_.flatten](https://github.com/lodash/lodash/tree/3.10.1/doc#_flattenarray-isdeep)
* [\_.flattenDeep](https://github.com/lodash/lodash/tree/3.10.1/doc#_flattendeeparray)
* [\_.head -> first](https://github.com/lodash/lodash/tree/3.10.1/doc#_firstarray)
* [\_.indexOf](https://github.com/lodash/lodash/tree/3.10.1/doc#_indexofarray-value-fromindex0)
* [\_.initial](https://github.com/lodash/lodash/tree/3.10.1/doc#_initialarray)
* [\_.intersection](https://github.com/lodash/lodash/tree/3.10.1/doc#_intersectionarrays)
* [\_.last](https://github.com/lodash/lodash/tree/3.10.1/doc#_lastarray)
* [\_.lastIndexOf](https://github.com/lodash/lodash/tree/3.10.1/doc#_lastindexofarray-value-fromindexarraylength-1)
* [\_.object -> zipObject](https://github.com/lodash/lodash/tree/3.10.1/doc#_zipobjectprops-values)
* [\_.pull](https://github.com/lodash/lodash/tree/3.10.1/doc#_pullarray-values)
* [\_.pullAt](https://github.com/lodash/lodash/tree/3.10.1/doc#_pullatarray-indexes)
* [\_.remove](https://github.com/lodash/lodash/tree/3.10.1/doc#_removearray-predicate_identity-thisarg)
* [\_.rest](https://github.com/lodash/lodash/tree/3.10.1/doc#_restarray)
* [\_.slice](https://github.com/lodash/lodash/tree/3.10.1/doc#_slicearray-start0-endarraylength)
* [\_.sortedIndex](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortedindexarray-value-iteratee_identity-thisarg)
* [\_.sortedLastIndex](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortedlastindexarray-value-iteratee_identity-thisarg)
* [\_.tail -> rest](https://github.com/lodash/lodash/tree/3.10.1/doc#_restarray)
* [\_.take](https://github.com/lodash/lodash/tree/3.10.1/doc#_takearray-n1)
* [\_.takeRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_takerightarray-n1)
* [\_.takeRightWhile](https://github.com/lodash/lodash/tree/3.10.1/doc#_takerightwhilearray-predicate_identity-thisarg)
* [\_.takeWhile](https://github.com/lodash/lodash/tree/3.10.1/doc#_takewhilearray-predicate_identity-thisarg)
* [\_.union](https://github.com/lodash/lodash/tree/3.10.1/doc#_unionarrays)
* [\_.uniq](https://github.com/lodash/lodash/tree/3.10.1/doc#_uniqarray-issorted-iteratee-thisarg)
* [\_.unique -> uniq](https://github.com/lodash/lodash/tree/3.10.1/doc#_uniqarray-issorted-iteratee-thisarg)
* [\_.unzip](https://github.com/lodash/lodash/tree/3.10.1/doc#_unziparray)
* [\_.unzipWith](https://github.com/lodash/lodash/tree/3.10.1/doc#_unzipwitharray-iteratee-thisarg)
* [\_.without](https://github.com/lodash/lodash/tree/3.10.1/doc#_withoutarray-values)
* [\_.xor](https://github.com/lodash/lodash/tree/3.10.1/doc#_xorarrays)
* [\_.zip](https://github.com/lodash/lodash/tree/3.10.1/doc#_ziparrays)
* [\_.zipObject](https://github.com/lodash/lodash/tree/3.10.1/doc#_zipobjectprops-values)
* [\_.zipWith](https://github.com/lodash/lodash/tree/3.10.1/doc#_zipwitharrays-iteratee-thisarg)

**Chain**

* [\_](https://github.com/lodash/lodash/tree/3.10.1/doc#_value)
* [\_.chain](https://github.com/lodash/lodash/tree/3.10.1/doc#_chainvalue)
* [\_.tap](https://github.com/lodash/lodash/tree/3.10.1/doc#_tapvalue-interceptor-thisarg)
* [\_.thru](https://github.com/lodash/lodash/tree/3.10.1/doc#_thruvalue-interceptor-thisarg)
* [\_.prototype.chain](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypechain)
* [\_.prototype.commit](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypecommit)
* [\_.prototype.concat](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypeconcatvalues)
* [\_.prototype.plant](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypeplant)
* [\_.prototype.reverse](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypereverse)
* [\_.prototype.run -> value](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypevalue)
* [\_.prototype.toJSON -> value](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypevalue)
* [\_.prototype.toString](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypetostring)
* [\_.prototype.value](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypevalue)
* [\_.prototype.valueOf -> value](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypevalue)

**Collection**

* [\_.all -> every](https://github.com/lodash/lodash/tree/3.10.1/doc#_everycollection-predicate_identity-thisarg)
* [\_.any -> some](https://github.com/lodash/lodash/tree/3.10.1/doc#_somecollection-predicate_identity-thisarg)
* [\_.at](https://github.com/lodash/lodash/tree/3.10.1/doc#_atcollection-props)
* [\_.collect -> map](https://github.com/lodash/lodash/tree/3.10.1/doc#_mapcollection-iteratee_identity-thisarg)
* [\_.contains -> includes](https://github.com/lodash/lodash/tree/3.10.1/doc#_includescollection-target-fromindex0)
* [\_.countBy](https://github.com/lodash/lodash/tree/3.10.1/doc#_countbycollection-iteratee_identity-thisarg)
* [\_.detect -> find](https://github.com/lodash/lodash/tree/3.10.1/doc#_findcollection-predicate_identity-thisarg)
* [\_.each -> forEach](https://github.com/lodash/lodash/tree/3.10.1/doc#_foreachcollection-iteratee_identity-thisarg)
* [\_.eachRight -> forEachRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_foreachrightcollection-iteratee_identity-thisarg)
* [\_.every](https://github.com/lodash/lodash/tree/3.10.1/doc#_everycollection-predicate_identity-thisarg)
* [\_.filter](https://github.com/lodash/lodash/tree/3.10.1/doc#_filtercollection-predicate_identity-thisarg)
* [\_.find](https://github.com/lodash/lodash/tree/3.10.1/doc#_findcollection-predicate_identity-thisarg)
* [\_.findLast](https://github.com/lodash/lodash/tree/3.10.1/doc#_findlastcollection-predicate_identity-thisarg)
* [\_.findWhere](https://github.com/lodash/lodash/tree/3.10.1/doc#_findwherecollection-source)
* [\_.foldl -> reduce](https://github.com/lodash/lodash/tree/3.10.1/doc#_reducecollection-iteratee_identity-accumulator-thisarg)
* [\_.foldr -> reduceRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_reducerightcollection-iteratee_identity-accumulator-thisarg)
* [\_.forEach](https://github.com/lodash/lodash/tree/3.10.1/doc#_foreachcollection-iteratee_identity-thisarg)
* [\_.forEachRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_foreachrightcollection-iteratee_identity-thisarg)
* [\_.groupBy](https://github.com/lodash/lodash/tree/3.10.1/doc#_groupbycollection-iteratee_identity-thisarg)
* [\_.include -> includes](https://github.com/lodash/lodash/tree/3.10.1/doc#_includescollection-target-fromindex0)
* [\_.includes](https://github.com/lodash/lodash/tree/3.10.1/doc#_includescollection-target-fromindex0)
* [\_.indexBy](https://github.com/lodash/lodash/tree/3.10.1/doc#_indexbycollection-iteratee_identity-thisarg)
* [\_.inject -> reduce](https://github.com/lodash/lodash/tree/3.10.1/doc#_reducecollection-iteratee_identity-accumulator-thisarg)
* [\_.invoke](https://github.com/lodash/lodash/tree/3.10.1/doc#_invokecollection-path-args)
* [\_.map](https://github.com/lodash/lodash/tree/3.10.1/doc#_mapcollection-iteratee_identity-thisarg)
* [\_.partition](https://github.com/lodash/lodash/tree/3.10.1/doc#_partitioncollection-predicate_identity-thisarg)
* [\_.pluck](https://github.com/lodash/lodash/tree/3.10.1/doc#_pluckcollection-path)
* [\_.reduce](https://github.com/lodash/lodash/tree/3.10.1/doc#_reducecollection-iteratee_identity-accumulator-thisarg)
* [\_.reduceRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_reducerightcollection-iteratee_identity-accumulator-thisarg)
* [\_.reject](https://github.com/lodash/lodash/tree/3.10.1/doc#_rejectcollection-predicate_identity-thisarg)
* [\_.sample](https://github.com/lodash/lodash/tree/3.10.1/doc#_samplecollection-n)
* [\_.select -> filter](https://github.com/lodash/lodash/tree/3.10.1/doc#_filtercollection-predicate_identity-thisarg)
* [\_.shuffle](https://github.com/lodash/lodash/tree/3.10.1/doc#_shufflecollection)
* [\_.size](https://github.com/lodash/lodash/tree/3.10.1/doc#_sizecollection)
* [\_.some](https://github.com/lodash/lodash/tree/3.10.1/doc#_somecollection-predicate_identity-thisarg)
* [\_.sortBy](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortbycollection-iteratee_identity-thisarg)
* [\_.sortByAll](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortbyallcollection-iteratees)
* [\_.sortByOrder](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortbyordercollection-iteratees-orders)
* [\_.where](https://github.com/lodash/lodash/tree/3.10.1/doc#_wherecollection-source)

**Date**

* [\_.now](https://github.com/lodash/lodash/tree/3.10.1/doc#_now)

**Function**

* [\_.after](https://github.com/lodash/lodash/tree/3.10.1/doc#_aftern-func)
* [\_.ary](https://github.com/lodash/lodash/tree/3.10.1/doc#_aryfunc-nfunclength)
* [\_.backflow -> flowRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_flowrightfuncs)
* [\_.before](https://github.com/lodash/lodash/tree/3.10.1/doc#_beforen-func)
* [\_.bind](https://github.com/lodash/lodash/tree/3.10.1/doc#_bindfunc-thisarg-partials)
* [\_.bindAll](https://github.com/lodash/lodash/tree/3.10.1/doc#_bindallobject-methodnames)
* [\_.bindKey](https://github.com/lodash/lodash/tree/3.10.1/doc#_bindkeyobject-key-partials)
* [\_.compose -> flowRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_flowrightfuncs)
* [\_.curry](https://github.com/lodash/lodash/tree/3.10.1/doc#_curryfunc-arityfunclength)
* [\_.curryRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_curryrightfunc-arityfunclength)
* [\_.debounce](https://github.com/lodash/lodash/tree/3.10.1/doc#_debouncefunc-wait0-options)
* [\_.defer](https://github.com/lodash/lodash/tree/3.10.1/doc#_deferfunc-args)
* [\_.delay](https://github.com/lodash/lodash/tree/3.10.1/doc#_delayfunc-wait-args)
* [\_.flow](https://github.com/lodash/lodash/tree/3.10.1/doc#_flowfuncs)
* [\_.flowRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_flowrightfuncs)
* [\_.memoize](https://github.com/lodash/lodash/tree/3.10.1/doc#_memoizefunc-resolver)
* [\_.modArgs](https://github.com/lodash/lodash/tree/3.10.1/doc#_modargsfunc-transforms)
* [\_.negate](https://github.com/lodash/lodash/tree/3.10.1/doc#_negatepredicate)
* [\_.once](https://github.com/lodash/lodash/tree/3.10.1/doc#_oncefunc)
* [\_.partial](https://github.com/lodash/lodash/tree/3.10.1/doc#_partialfunc-partials)
* [\_.partialRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_partialrightfunc-partials)
* [\_.rearg](https://github.com/lodash/lodash/tree/3.10.1/doc#_reargfunc-indexes)
* [\_.restParam](https://github.com/lodash/lodash/tree/3.10.1/doc#_restparamfunc-startfunclength-1)
* [\_.spread](https://github.com/lodash/lodash/tree/3.10.1/doc#_spreadfunc)
* [\_.throttle](https://github.com/lodash/lodash/tree/3.10.1/doc#_throttlefunc-wait0-options)
* [\_.wrap](https://github.com/lodash/lodash/tree/3.10.1/doc#_wrapvalue-wrapper)

**Lang**

* [\_.clone](https://github.com/lodash/lodash/tree/3.10.1/doc#_clonevalue-isdeep-customizer-thisarg)
* [\_.cloneDeep](https://github.com/lodash/lodash/tree/3.10.1/doc#_clonedeepvalue-customizer-thisarg)
* [\_.eq -> isEqual](https://github.com/lodash/lodash/tree/3.10.1/doc#_isequalvalue-other-customizer-thisarg)
* [\_.gt](https://github.com/lodash/lodash/tree/3.10.1/doc#_gtvalue-other)
* [\_.gte](https://github.com/lodash/lodash/tree/3.10.1/doc#_gtevalue-other)
* [\_.isArguments](https://github.com/lodash/lodash/tree/3.10.1/doc#_isargumentsvalue)
* [\_.isArray](https://github.com/lodash/lodash/tree/3.10.1/doc#_isarrayvalue)
* [\_.isBoolean](https://github.com/lodash/lodash/tree/3.10.1/doc#_isbooleanvalue)
* [\_.isDate](https://github.com/lodash/lodash/tree/3.10.1/doc#_isdatevalue)
* [\_.isElement](https://github.com/lodash/lodash/tree/3.10.1/doc#_iselementvalue)
* [\_.isEmpty](https://github.com/lodash/lodash/tree/3.10.1/doc#_isemptyvalue)
* [\_.isEqual](https://github.com/lodash/lodash/tree/3.10.1/doc#_isequalvalue-other-customizer-thisarg)
* [\_.isError](https://github.com/lodash/lodash/tree/3.10.1/doc#_iserrorvalue)
* [\_.isFinite](https://github.com/lodash/lodash/tree/3.10.1/doc#_isfinitevalue)
* [\_.isFunction](https://github.com/lodash/lodash/tree/3.10.1/doc#_isfunctionvalue)
* [\_.isMatch](https://github.com/lodash/lodash/tree/3.10.1/doc#_ismatchobject-source-customizer-thisarg)
* [\_.isNaN](https://github.com/lodash/lodash/tree/3.10.1/doc#_isnanvalue)
* [\_.isNative](https://github.com/lodash/lodash/tree/3.10.1/doc#_isnativevalue)
* [\_.isNull](https://github.com/lodash/lodash/tree/3.10.1/doc#_isnullvalue)
* [\_.isNumber](https://github.com/lodash/lodash/tree/3.10.1/doc#_isnumbervalue)
* [\_.isObject](https://github.com/lodash/lodash/tree/3.10.1/doc#_isobjectvalue)
* [\_.isPlainObject](https://github.com/lodash/lodash/tree/3.10.1/doc#_isplainobjectvalue)
* [\_.isRegExp](https://github.com/lodash/lodash/tree/3.10.1/doc#_isregexpvalue)
* [\_.isString](https://github.com/lodash/lodash/tree/3.10.1/doc#_isstringvalue)
* [\_.isTypedArray](https://github.com/lodash/lodash/tree/3.10.1/doc#_istypedarrayvalue)
* [\_.isUndefined](https://github.com/lodash/lodash/tree/3.10.1/doc#_isundefinedvalue)
* [\_.lt](https://github.com/lodash/lodash/tree/3.10.1/doc#_ltvalue-other)
* [\_.lte](https://github.com/lodash/lodash/tree/3.10.1/doc#_ltevalue-other)
* [\_.toArray](https://github.com/lodash/lodash/tree/3.10.1/doc#_toarrayvalue)
* [\_.toPlainObject](https://github.com/lodash/lodash/tree/3.10.1/doc#_toplainobjectvalue)

**Math**

* [\_.add](https://github.com/lodash/lodash/tree/3.10.1/doc#_addaugend-addend)
* [\_.ceil](https://github.com/lodash/lodash/tree/3.10.1/doc#_ceiln-precision0)
* [\_.floor](https://github.com/lodash/lodash/tree/3.10.1/doc#_floorn-precision0)
* [\_.max](https://github.com/lodash/lodash/tree/3.10.1/doc#_maxcollection-iteratee-thisarg)
* [\_.min](https://github.com/lodash/lodash/tree/3.10.1/doc#_mincollection-iteratee-thisarg)
* [\_.round](https://github.com/lodash/lodash/tree/3.10.1/doc#_roundn-precision0)
* [\_.sum](https://github.com/lodash/lodash/tree/3.10.1/doc#_sumcollection-iteratee-thisarg)

**Number**

* [\_.inRange](https://github.com/lodash/lodash/tree/3.10.1/doc#_inrangen-start0-end)
* [\_.random](https://github.com/lodash/lodash/tree/3.10.1/doc#_randommin0-max1-floating)

**Object**

* [\_.assign](https://github.com/lodash/lodash/tree/3.10.1/doc#_assignobject-sources-customizer-thisarg)
* [\_.create](https://github.com/lodash/lodash/tree/3.10.1/doc#_createprototype-properties)
* [\_.defaults](https://github.com/lodash/lodash/tree/3.10.1/doc#_defaultsobject-sources)
* [\_.defaultsDeep](https://github.com/lodash/lodash/tree/3.10.1/doc#_defaultsdeepobject-sources)
* [\_.extend -> assign](https://github.com/lodash/lodash/tree/3.10.1/doc#_assignobject-sources-customizer-thisarg)
* [\_.findKey](https://github.com/lodash/lodash/tree/3.10.1/doc#_findkeyobject-predicate_identity-thisarg)
* [\_.findLastKey](https://github.com/lodash/lodash/tree/3.10.1/doc#_findlastkeyobject-predicate_identity-thisarg)
* [\_.forIn](https://github.com/lodash/lodash/tree/3.10.1/doc#_forinobject-iteratee_identity-thisarg)
* [\_.forInRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_forinrightobject-iteratee_identity-thisarg)
* [\_.forOwn](https://github.com/lodash/lodash/tree/3.10.1/doc#_forownobject-iteratee_identity-thisarg)
* [\_.forOwnRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_forownrightobject-iteratee_identity-thisarg)
* [\_.functions](https://github.com/lodash/lodash/tree/3.10.1/doc#_functionsobject)
* [\_.get](https://github.com/lodash/lodash/tree/3.10.1/doc#_getobject-path-defaultvalue)
* [\_.has](https://github.com/lodash/lodash/tree/3.10.1/doc#_hasobject-path)
* [\_.invert](https://github.com/lodash/lodash/tree/3.10.1/doc#_invertobject-multivalue)
* [\_.keys](https://github.com/lodash/lodash/tree/3.10.1/doc#_keysobject)
* [\_.keysIn](https://github.com/lodash/lodash/tree/3.10.1/doc#_keysinobject)
* [\_.mapKeys](https://github.com/lodash/lodash/tree/3.10.1/doc#_mapkeysobject-iteratee_identity-thisarg)
* [\_.mapValues](https://github.com/lodash/lodash/tree/3.10.1/doc#_mapvaluesobject-iteratee_identity-thisarg)
* [\_.merge](https://github.com/lodash/lodash/tree/3.10.1/doc#_mergeobject-sources-customizer-thisarg)
* [\_.methods -> functions](https://github.com/lodash/lodash/tree/3.10.1/doc#_functionsobject)
* [\_.omit](https://github.com/lodash/lodash/tree/3.10.1/doc#_omitobject-predicate-thisarg)
* [\_.pairs](https://github.com/lodash/lodash/tree/3.10.1/doc#_pairsobject)
* [\_.pick](https://github.com/lodash/lodash/tree/3.10.1/doc#_pickobject-predicate-thisarg)
* [\_.result](https://github.com/lodash/lodash/tree/3.10.1/doc#_resultobject-path-defaultvalue)
* [\_.set](https://github.com/lodash/lodash/tree/3.10.1/doc#_setobject-path-value)
* [\_.transform](https://github.com/lodash/lodash/tree/3.10.1/doc#_transformobject-iteratee_identity-accumulator-thisarg)
* [\_.values](https://github.com/lodash/lodash/tree/3.10.1/doc#_valuesobject)
* [\_.valuesIn](https://github.com/lodash/lodash/tree/3.10.1/doc#_valuesinobject)

**String**

* [\_.camelCase](https://github.com/lodash/lodash/tree/3.10.1/doc#_camelcasestring)
* [\_.capitalize](https://github.com/lodash/lodash/tree/3.10.1/doc#_capitalizestring)
* [\_.deburr](https://github.com/lodash/lodash/tree/3.10.1/doc#_deburrstring)
* [\_.endsWith](https://github.com/lodash/lodash/tree/3.10.1/doc#_endswithstring-target-positionstringlength)
* [\_.escape](https://github.com/lodash/lodash/tree/3.10.1/doc#_escapestring)
* [\_.escapeRegExp](https://github.com/lodash/lodash/tree/3.10.1/doc#_escaperegexpstring)
* [\_.kebabCase](https://github.com/lodash/lodash/tree/3.10.1/doc#_kebabcasestring)
* [\_.pad](https://github.com/lodash/lodash/tree/3.10.1/doc#_padstring-length0-chars)
* [\_.padLeft](https://github.com/lodash/lodash/tree/3.10.1/doc#_padleftstring-length0-chars)
* [\_.padRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_padrightstring-length0-chars)
* [\_.parseInt](https://github.com/lodash/lodash/tree/3.10.1/doc#_parseintstring-radix)
* [\_.repeat](https://github.com/lodash/lodash/tree/3.10.1/doc#_repeatstring-n0)
* [\_.snakeCase](https://github.com/lodash/lodash/tree/3.10.1/doc#_snakecasestring)
* [\_.startCase](https://github.com/lodash/lodash/tree/3.10.1/doc#_startcasestring)
* [\_.startsWith](https://github.com/lodash/lodash/tree/3.10.1/doc#_startswithstring-target-position0)
* [\_.template](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatestring-options)
* [\_.trim](https://github.com/lodash/lodash/tree/3.10.1/doc#_trimstring-charswhitespace)
* [\_.trimLeft](https://github.com/lodash/lodash/tree/3.10.1/doc#_trimleftstring-charswhitespace)
* [\_.trimRight](https://github.com/lodash/lodash/tree/3.10.1/doc#_trimrightstring-charswhitespace)
* [\_.trunc](https://github.com/lodash/lodash/tree/3.10.1/doc#_truncstring-options-optionslength30-optionsomission-optionsseparator)
* [\_.unescape](https://github.com/lodash/lodash/tree/3.10.1/doc#_unescapestring)
* [\_.words](https://github.com/lodash/lodash/tree/3.10.1/doc#_wordsstring-pattern)

**Utility**

* [\_.attempt](https://github.com/lodash/lodash/tree/3.10.1/doc#_attemptfunc)
* [\_.callback](https://github.com/lodash/lodash/tree/3.10.1/doc#_callbackfunc_identity-thisarg)
* [\_.constant](https://github.com/lodash/lodash/tree/3.10.1/doc#_constantvalue)
* [\_.identity](https://github.com/lodash/lodash/tree/3.10.1/doc#_identityvalue)
* [\_.iteratee -> callback](https://github.com/lodash/lodash/tree/3.10.1/doc#_callbackfunc_identity-thisarg)
* [\_.matches](https://github.com/lodash/lodash/tree/3.10.1/doc#_matchessource)
* [\_.matchesProperty](https://github.com/lodash/lodash/tree/3.10.1/doc#_matchespropertypath-srcvalue)
* [\_.method](https://github.com/lodash/lodash/tree/3.10.1/doc#_methodpath-args)
* [\_.methodOf](https://github.com/lodash/lodash/tree/3.10.1/doc#_methodofobject-args)
* [\_.mixin](https://github.com/lodash/lodash/tree/3.10.1/doc#_mixinobjectlodash-source-options)
* [\_.noConflict](https://github.com/lodash/lodash/tree/3.10.1/doc#_noconflict)
* [\_.noop](https://github.com/lodash/lodash/tree/3.10.1/doc#_noop)
* [\_.property](https://github.com/lodash/lodash/tree/3.10.1/doc#_propertypath)
* [\_.propertyOf](https://github.com/lodash/lodash/tree/3.10.1/doc#_propertyofobject)
* [\_.range](https://github.com/lodash/lodash/tree/3.10.1/doc#_rangestart0-end-step1)
* [\_.runInContext](https://github.com/lodash/lodash/tree/3.10.1/doc#_runincontextcontextroot)
* [\_.times](https://github.com/lodash/lodash/tree/3.10.1/doc#_timesn-iteratee_identity-thisarg)
* [\_.uniqueId](https://github.com/lodash/lodash/tree/3.10.1/doc#_uniqueidprefix)

**Methods**

* [\_.templateSettings.imports.\_](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsimports_)

**Properties**

* [\_.VERSION](https://github.com/lodash/lodash/tree/3.10.1/doc#_version)
* [\_.support](https://github.com/lodash/lodash/tree/3.10.1/doc#_support)
* [\_.support.enumErrorProps](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportenumerrorprops)
* [\_.support.enumPrototypes](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportenumprototypes)
* [\_.support.nonEnumShadows](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportnonenumshadows)
* [\_.support.ownLast](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportownlast)
* [\_.support.spliceObjects](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportspliceobjects)
* [\_.support.unindexedChars](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportunindexedchars)
* [\_.templateSettings](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettings)
* [\_.templateSettings.escape](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsescape)
* [\_.templateSettings.evaluate](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsevaluate)
* [\_.templateSettings.imports](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsimports)
* [\_.templateSettings.interpolate](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsinterpolate)
* [\_.templateSettings.variable](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsvariable)

**“Array” Methods**

**\_.chunk(array, [size=1])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_chunkarray-size1) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4624) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.chunk)

Creates an array of elements split into groups the length of size. If collection can't be split evenly, the final chunk will be the remaining elements.

**Arguments**

1. array *(Array)*: The array to process.
2. [size=1] *(number)*: The length of each chunk.

**Returns**

*(Array)*: Returns the new array containing chunks.

**Example**

\_.chunk(['a', 'b', 'c', 'd'], 2);

// => [['a', 'b'], ['c', 'd']]

\_.chunk(['a', 'b', 'c', 'd'], 3);

// => [['a', 'b', 'c'], ['d']]

**\_.compact(array)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_compactarray) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4655) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.compact)

Creates an array with all falsey values removed. The values false, null, 0, "", undefined, and NaN are falsey.

**Arguments**

1. array *(Array)*: The array to compact.

**Returns**

*(Array)*: Returns the new array of filtered values.

**Example**

\_.compact([0, 1, false, 2, '', 3]);

// => [1, 2, 3]

**\_.difference(array, [values])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_differencearray-values) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4686) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.difference)

Creates an array of unique array values not included in the other provided arrays using [SameValueZero](http://ecma-international.org/ecma-262/6.0/#sec-samevaluezero) for equality comparisons.

**Arguments**

1. array *(Array)*: The array to inspect.
2. [values] *(...Array)*: The arrays of values to exclude.

**Returns**

*(Array)*: Returns the new array of filtered values.

**Example**

\_.difference([1, 2, 3], [4, 2]);

// => [1, 3]

**\_.drop(array, [n=1])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_droparray-n1) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4716) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.drop)

Creates a slice of array with n elements dropped from the beginning.

**Arguments**

1. array *(Array)*: The array to query.
2. [n=1] *(number)*: The number of elements to drop.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.drop([1, 2, 3]);

// => [2, 3]

\_.drop([1, 2, 3], 2);

// => [3]

\_.drop([1, 2, 3], 5);

// => []

\_.drop([1, 2, 3], 0);

// => [1, 2, 3]

**\_.dropRight(array, [n=1])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_droprightarray-n1) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4751) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.dropright)

Creates a slice of array with n elements dropped from the end.

**Arguments**

1. array *(Array)*: The array to query.
2. [n=1] *(number)*: The number of elements to drop.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.dropRight([1, 2, 3]);

// => [1, 2]

\_.dropRight([1, 2, 3], 2);

// => [1]

\_.dropRight([1, 2, 3], 5);

// => []

\_.dropRight([1, 2, 3], 0);

// => [1, 2, 3]

**\_.dropRightWhile(array, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_droprightwhilearray-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4812) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.droprightwhile)

Creates a slice of array excluding elements dropped from the end. Elements are dropped until predicate returns falsey. The predicate is bound to thisArg and invoked with three arguments: (value, index, array).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that match the properties of the given object, else false.

**Arguments**

1. array *(Array)*: The array to query.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.dropRightWhile([1, 2, 3], function(n) {

return n > 1;

});

// => [1]

var users = [

{ 'user': 'barney', 'active': true },

{ 'user': 'fred', 'active': false },

{ 'user': 'pebbles', 'active': false }

];

// using the `\_.matches` callback shorthand

\_.pluck(\_.dropRightWhile(users, { 'user': 'pebbles', 'active': false }), 'user');

// => ['barney', 'fred']

// using the `\_.matchesProperty` callback shorthand

\_.pluck(\_.dropRightWhile(users, 'active', false), 'user');

// => ['barney']

// using the `\_.property` callback shorthand

\_.pluck(\_.dropRightWhile(users, 'active'), 'user');

// => ['barney', 'fred', 'pebbles']

**\_.dropWhile(array, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_dropwhilearray-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4867) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.dropwhile)

Creates a slice of array excluding elements dropped from the beginning. Elements are dropped until predicate returns falsey. The predicate is bound to thisArg and invoked with three arguments: (value, index, array).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. array *(Array)*: The array to query.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.dropWhile([1, 2, 3], function(n) {

return n < 3;

});

// => [3]

var users = [

{ 'user': 'barney', 'active': false },

{ 'user': 'fred', 'active': false },

{ 'user': 'pebbles', 'active': true }

];

// using the `\_.matches` callback shorthand

\_.pluck(\_.dropWhile(users, { 'user': 'barney', 'active': false }), 'user');

// => ['fred', 'pebbles']

// using the `\_.matchesProperty` callback shorthand

\_.pluck(\_.dropWhile(users, 'active', false), 'user');

// => ['pebbles']

// using the `\_.property` callback shorthand

\_.pluck(\_.dropWhile(users, 'active'), 'user');

// => ['barney', 'fred', 'pebbles']

**\_.fill(array, value, [start=0], [end=array.length])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_fillarray-value-start0-endarraylength) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4901) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.fill)

Fills elements of array with value from start up to, but not including, end.   
  
**Note:** This method mutates array.

**Arguments**

1. array *(Array)*: The array to fill.
2. value *(\*)*: The value to fill array with.
3. [start=0] *(number)*: The start position.
4. [end=array.length] *(number)*: The end position.

**Returns**

*(Array)*: Returns array.

**Example**

var array = [1, 2, 3];

\_.fill(array, 'a');

console.log(array);

// => ['a', 'a', 'a']

\_.fill(Array(3), 2);

// => [2, 2, 2]

\_.fill([4, 6, 8], '\*', 1, 2);

// => [4, '\*', 8]

**\_.findIndex(array, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_findindexarray-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L4961) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.findindex)

This method is like \_.find except that it returns the index of the first element predicate returns truthy for instead of the element itself.   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. array *(Array)*: The array to search.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(number)*: Returns the index of the found element, else -1.

**Example**

var users = [

{ 'user': 'barney', 'active': false },

{ 'user': 'fred', 'active': false },

{ 'user': 'pebbles', 'active': true }

];

\_.findIndex(users, function(chr) {

return chr.user == 'barney';

});

// => 0

// using the `\_.matches` callback shorthand

\_.findIndex(users, { 'user': 'fred', 'active': false });

// => 1

// using the `\_.matchesProperty` callback shorthand

\_.findIndex(users, 'active', false);

// => 0

// using the `\_.property` callback shorthand

\_.findIndex(users, 'active');

// => 2

**\_.findLastIndex(array, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_findlastindexarray-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5011) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.findlastindex)

This method is like \_.findIndex except that it iterates over elements of collection from right to left.   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. array *(Array)*: The array to search.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(number)*: Returns the index of the found element, else -1.

**Example**

var users = [

{ 'user': 'barney', 'active': true },

{ 'user': 'fred', 'active': false },

{ 'user': 'pebbles', 'active': false }

];

\_.findLastIndex(users, function(chr) {

return chr.user == 'pebbles';

});

// => 2

// using the `\_.matches` callback shorthand

\_.findLastIndex(users, { 'user': 'barney', 'active': true });

// => 0

// using the `\_.matchesProperty` callback shorthand

\_.findLastIndex(users, 'active', false);

// => 2

// using the `\_.property` callback shorthand

\_.findLastIndex(users, 'active');

// => 0

**\_.first(array)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_firstarray) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5030) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.first)

Gets the first element of array.

**Aliases**

*\_.head*

**Arguments**

1. array *(Array)*: The array to query.

**Returns**

*(\*)*: Returns the first element of array.

**Example**

\_.first([1, 2, 3]);

// => 1

\_.first([]);

// => undefined

**\_.flatten(array, [isDeep])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_flattenarray-isdeep) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5054) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.flatten)

Flattens a nested array. If isDeep is true the array is recursively flattened, otherwise it's only flattened a single level.

**Arguments**

1. array *(Array)*: The array to flatten.
2. [isDeep] *(boolean)*: Specify a deep flatten.

**Returns**

*(Array)*: Returns the new flattened array.

**Example**

\_.flatten([1, [2, 3, [4]]]);

// => [1, 2, 3, [4]]

// using `isDeep`

\_.flatten([1, [2, 3, [4]]], true);

// => [1, 2, 3, 4]

**\_.flattenDeep(array)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_flattendeeparray) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5075) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.flattendeep)

Recursively flattens a nested array.

**Arguments**

1. array *(Array)*: The array to recursively flatten.

**Returns**

*(Array)*: Returns the new flattened array.

**Example**

\_.flattenDeep([1, [2, 3, [4]]]);

// => [1, 2, 3, 4]

**\_.indexOf(array, value, [fromIndex=0])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_indexofarray-value-fromindex0) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5108) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.indexof)

Gets the index at which the first occurrence of value is found in array using [SameValueZero](http://ecma-international.org/ecma-262/6.0/#sec-samevaluezero) for equality comparisons. IffromIndex is negative, it's used as the offset from the end of array. If array is sorted providing true for fromIndexperforms a faster binary search.

**Arguments**

1. array *(Array)*: The array to search.
2. value *(\*)*: The value to search for.
3. [fromIndex=0] *(boolean|number)*: The index to search from or true to perform a binary search on a sorted array.

**Returns**

*(number)*: Returns the index of the matched value, else -1.

**Example**

\_.indexOf([1, 2, 1, 2], 2);

// => 1

// using `fromIndex`

\_.indexOf([1, 2, 1, 2], 2, 2);

// => 3

// performing a binary search

\_.indexOf([1, 1, 2, 2], 2, true);

// => 2

**\_.initial(array)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_initialarray) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5139) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.initial)

Gets all but the last element of array.

**Arguments**

1. array *(Array)*: The array to query.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.initial([1, 2, 3]);

// => [1, 2]

**\_.intersection([arrays])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_intersectionarrays) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5157) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.intersection)

Creates an array of unique values that are included in all of the provided arrays using [SameValueZero](http://ecma-international.org/ecma-262/6.0/#sec-samevaluezero) for equality comparisons.

**Arguments**

1. [arrays] *(...Array)*: The arrays to inspect.

**Returns**

*(Array)*: Returns the new array of shared values.

**Example**

\_.intersection([1, 2], [4, 2], [2, 1]);

// => [2]

**\_.last(array)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_lastarray) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5207) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.last)

Gets the last element of array.

**Arguments**

1. array *(Array)*: The array to query.

**Returns**

*(\*)*: Returns the last element of array.

**Example**

\_.last([1, 2, 3]);

// => 3

**\_.lastIndexOf(array, value, [fromIndex=array.length-1])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_lastindexofarray-value-fromindexarraylength-1) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5237) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.lastindexof)

This method is like \_.indexOf except that it iterates over elements of array from right to left.

**Arguments**

1. array *(Array)*: The array to search.
2. value *(\*)*: The value to search for.
3. [fromIndex=array.length-1] *(boolean|number)*: The index to search from or true to perform a binary search on a sorted array.

**Returns**

*(number)*: Returns the index of the matched value, else -1.

**Example**

\_.lastIndexOf([1, 2, 1, 2], 2);

// => 3

// using `fromIndex`

\_.lastIndexOf([1, 2, 1, 2], 2, 2);

// => 1

// performing a binary search

\_.lastIndexOf([1, 1, 2, 2], 2, true);

// => 3

**\_.pull(array, [values])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_pullarray-values) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5285) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.pull)

Removes all provided values from array using [SameValueZero](http://ecma-international.org/ecma-262/6.0/#sec-samevaluezero) for equality comparisons.   
  
**Note:** Unlike \_.without, this method mutates array.

**Arguments**

1. array *(Array)*: The array to modify.
2. [values] *(...\*)*: The values to remove.

**Returns**

*(Array)*: Returns array.

**Example**

var array = [1, 2, 3, 1, 2, 3];

\_.pull(array, 2, 3);

console.log(array);

// => [1, 1]

**\_.pullAt(array, [indexes])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_pullatarray-indexes) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5332) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.pullat)

Removes elements from array corresponding to the given indexes and returns an array of the removed elements. Indexes may be specified as an array of indexes or as individual arguments.   
  
**Note:** Unlike \_.at, this method mutates array.

**Arguments**

1. array *(Array)*: The array to modify.
2. [indexes] *(...(number|number[])*: The indexes of elements to remove, specified as individual indexes or arrays of indexes.

**Returns**

*(Array)*: Returns the new array of removed elements.

**Example**

var array = [5, 10, 15, 20];

var evens = \_.pullAt(array, 1, 3);

console.log(array);

// => [5, 15]

console.log(evens);

// => [10, 20]

**\_.remove(array, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_removearray-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5379) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.remove)

Removes all elements from array that predicate returns truthy for and returns an array of the removed elements. The predicate is bound to thisArg and invoked with three arguments: (value, index, array).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.   
  
**Note:** Unlike \_.filter, this method mutates array.

**Arguments**

1. array *(Array)*: The array to modify.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Array)*: Returns the new array of removed elements.

**Example**

var array = [1, 2, 3, 4];

var evens = \_.remove(array, function(n) {

return n % 2 == 0;

});

console.log(array);

// => [1, 3]

console.log(evens);

// => [2, 4]

**\_.rest(array)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_restarray) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5414) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.rest)

Gets all but the first element of array.

**Aliases**

*\_.tail*

**Arguments**

1. array *(Array)*: The array to query.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.rest([1, 2, 3]);

// => [2, 3]

**\_.slice(array, [start=0], [end=array.length])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_slicearray-start0-endarraylength) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5432) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.slice)

Creates a slice of array from start up to, but not including, end.   
  
**Note:** This method is used instead of Array#slice to support node lists in IE < 9 and to ensure dense arrays are returned.

**Arguments**

1. array *(Array)*: The array to slice.
2. [start=0] *(number)*: The start position.
3. [end=array.length] *(number)*: The end position.

**Returns**

*(Array)*: Returns the slice of array.

**\_.sortedIndex(array, value, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortedindexarray-value-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5492) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.sortedindex)

Uses a binary search to determine the lowest index at which value should be inserted into array in order to maintain its sort order. If an iteratee function is provided it's invoked for value and each element of array to compute their sort ranking. The iteratee is bound to thisArg and invoked with one argument; (value).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. array *(Array)*: The sorted array to inspect.
2. value *(\*)*: The value to evaluate.
3. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
4. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(number)*: Returns the index at which value should be inserted into array.

**Example**

\_.sortedIndex([30, 50], 40);

// => 1

\_.sortedIndex([4, 4, 5, 5], 5);

// => 2

var dict = { 'data': { 'thirty': 30, 'forty': 40, 'fifty': 50 } };

// using an iteratee function

\_.sortedIndex(['thirty', 'fifty'], 'forty', function(word) {

return this.data[word];

}, dict);

// => 1

// using the `\_.property` callback shorthand

\_.sortedIndex([{ 'x': 30 }, { 'x': 50 }], { 'x': 40 }, 'x');

// => 1

**\_.sortedLastIndex(array, value, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortedlastindexarray-value-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5514) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.sortedlastindex)

This method is like \_.sortedIndex except that it returns the highest index at which value should be inserted into array in order to maintain its sort order.

**Arguments**

1. array *(Array)*: The sorted array to inspect.
2. value *(\*)*: The value to evaluate.
3. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
4. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(number)*: Returns the index at which value should be inserted into array.

**Example**

\_.sortedLastIndex([4, 4, 5, 5], 5);

// => 4

**\_.take(array, [n=1])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_takearray-n1) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5540) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.take)

Creates a slice of array with n elements taken from the beginning.

**Arguments**

1. array *(Array)*: The array to query.
2. [n=1] *(number)*: The number of elements to take.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.take([1, 2, 3]);

// => [1]

\_.take([1, 2, 3], 2);

// => [1, 2]

\_.take([1, 2, 3], 5);

// => [1, 2, 3]

\_.take([1, 2, 3], 0);

// => []

**\_.takeRight(array, [n=1])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_takerightarray-n1) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5575) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.takeright)

Creates a slice of array with n elements taken from the end.

**Arguments**

1. array *(Array)*: The array to query.
2. [n=1] *(number)*: The number of elements to take.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.takeRight([1, 2, 3]);

// => [3]

\_.takeRight([1, 2, 3], 2);

// => [2, 3]

\_.takeRight([1, 2, 3], 5);

// => [1, 2, 3]

\_.takeRight([1, 2, 3], 0);

// => []

**\_.takeRightWhile(array, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_takerightwhilearray-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5636) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.takerightwhile)

Creates a slice of array with elements taken from the end. Elements are taken until predicate returns falsey. The predicate is bound to thisArg and invoked with three arguments: (value, index, array).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. array *(Array)*: The array to query.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.takeRightWhile([1, 2, 3], function(n) {

return n > 1;

});

// => [2, 3]

var users = [

{ 'user': 'barney', 'active': true },

{ 'user': 'fred', 'active': false },

{ 'user': 'pebbles', 'active': false }

];

// using the `\_.matches` callback shorthand

\_.pluck(\_.takeRightWhile(users, { 'user': 'pebbles', 'active': false }), 'user');

// => ['pebbles']

// using the `\_.matchesProperty` callback shorthand

\_.pluck(\_.takeRightWhile(users, 'active', false), 'user');

// => ['fred', 'pebbles']

// using the `\_.property` callback shorthand

\_.pluck(\_.takeRightWhile(users, 'active'), 'user');

// => []

**\_.takeWhile(array, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_takewhilearray-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5691) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.takewhile)

Creates a slice of array with elements taken from the beginning. Elements are taken until predicate returns falsey. The predicate is bound to thisArg and invoked with three arguments: (value, index, array).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. array *(Array)*: The array to query.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Array)*: Returns the slice of array.

**Example**

\_.takeWhile([1, 2, 3], function(n) {

return n < 3;

});

// => [1, 2]

var users = [

{ 'user': 'barney', 'active': false },

{ 'user': 'fred', 'active': false},

{ 'user': 'pebbles', 'active': true }

];

// using the `\_.matches` callback shorthand

\_.pluck(\_.takeWhile(users, { 'user': 'barney', 'active': false }), 'user');

// => ['barney']

// using the `\_.matchesProperty` callback shorthand

\_.pluck(\_.takeWhile(users, 'active', false), 'user');

// => ['barney', 'fred']

// using the `\_.property` callback shorthand

\_.pluck(\_.takeWhile(users, 'active'), 'user');

// => []

**\_.union([arrays])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_unionarrays) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5712) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.union)

Creates an array of unique values, in order, from all of the provided arrays using [SameValueZero](http://ecma-international.org/ecma-262/6.0/#sec-samevaluezero) for equality comparisons.

**Arguments**

1. [arrays] *(...Array)*: The arrays to inspect.

**Returns**

*(Array)*: Returns the new array of combined values.

**Example**

\_.union([1, 2], [4, 2], [2, 1]);

// => [1, 2, 4]

**\_.uniq(array, [isSorted], [iteratee], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_uniqarray-issorted-iteratee-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5765) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.uniq)

Creates a duplicate-free version of an array, using [SameValueZero](http://ecma-international.org/ecma-262/6.0/#sec-samevaluezero) for equality comparisons, in which only the first occurence of each element is kept. Providing true for isSorted performs a faster search algorithm for sorted arrays. If an iteratee function is provided it's invoked for each element in the array to generate the criterion by which uniqueness is computed. Theiteratee is bound to thisArg and invoked with three arguments: (value, index, array).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Aliases**

*\_.unique*

**Arguments**

1. array *(Array)*: The array to inspect.
2. [isSorted] *(boolean)*: Specify the array is sorted.
3. [iteratee] *(Function|Object|string)*: The function invoked per iteration.
4. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Array)*: Returns the new duplicate-value-free array.

**Example**

\_.uniq([2, 1, 2]);

// => [2, 1]

// using `isSorted`

\_.uniq([1, 1, 2], true);

// => [1, 2]

// using an iteratee function

\_.uniq([1, 2.5, 1.5, 2], function(n) {

return this.floor(n);

}, Math);

// => [1, 2.5]

// using the `\_.property` callback shorthand

\_.uniq([{ 'x': 1 }, { 'x': 2 }, { 'x': 1 }], 'x');

// => [{ 'x': 1 }, { 'x': 2 }]

**\_.unzip(array)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_unziparray) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5802) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.unzip)

This method is like \_.zip except that it accepts an array of grouped elements and creates an array regrouping the elements to their pre-zip configuration.

**Arguments**

1. array *(Array)*: The array of grouped elements to process.

**Returns**

*(Array)*: Returns the new array of regrouped elements.

**Example**

var zipped = \_.zip(['fred', 'barney'], [30, 40], [true, false]);

// => [['fred', 30, true], ['barney', 40, false]]

\_.unzip(zipped);

// => [['fred', 'barney'], [30, 40], [true, false]]

**\_.unzipWith(array, [iteratee], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_unzipwitharray-iteratee-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5842) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.unzipwith)

This method is like \_.unzip except that it accepts an iteratee to specify how regrouped values should be combined. Theiteratee is bound to thisArg and invoked with four arguments: (accumulator, value, index, group).

**Arguments**

1. array *(Array)*: The array of grouped elements to process.
2. [iteratee] *(Function)*: The function to combine regrouped values.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Array)*: Returns the new array of regrouped elements.

**Example**

var zipped = \_.zip([1, 2], [10, 20], [100, 200]);

// => [[1, 10, 100], [2, 20, 200]]

\_.unzipWith(zipped, \_.add);

// => [3, 30, 300]

**\_.without(array, [values])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_withoutarray-values) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5873) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.without)

Creates an array excluding all provided values using [SameValueZero](http://ecma-international.org/ecma-262/6.0/#sec-samevaluezero) for equality comparisons.

**Arguments**

1. array *(Array)*: The array to filter.
2. [values] *(...\*)*: The values to exclude.

**Returns**

*(Array)*: Returns the new array of filtered values.

**Example**

\_.without([1, 2, 1, 3], 1, 2);

// => [3]

**\_.xor([arrays])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_xorarrays) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5893) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.xor)

Creates an array of unique values that is the [symmetric difference](https://en.wikipedia.org/wiki/Symmetric_difference) of the provided arrays.

**Arguments**

1. [arrays] *(...Array)*: The arrays to inspect.

**Returns**

*(Array)*: Returns the new array of values.

**Example**

\_.xor([1, 2], [4, 2]);

// => [1, 4]

**\_.zip([arrays])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_ziparrays) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5923) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.zip)

Creates an array of grouped elements, the first of which contains the first elements of the given arrays, the second of which contains the second elements of the given arrays, and so on.

**Arguments**

1. [arrays] *(...Array)*: The arrays to process.

**Returns**

*(Array)*: Returns the new array of grouped elements.

**Example**

\_.zip(['fred', 'barney'], [30, 40], [true, false]);

// => [['fred', 30, true], ['barney', 40, false]]

**\_.zipObject(props, [values=[]])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_zipobjectprops-values) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5946) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.zipobject)

The inverse of \_.pairs; this method returns an object composed from arrays of property names and values. Provide either a single two dimensional array, e.g. [[key1, value1], [key2, value2]] or two arrays, one of property names and one of corresponding values.

**Aliases**

*\_.object*

**Arguments**

1. props *(Array)*: The property names.
2. [values=[]] *(Array)*: The property values.

**Returns**

*(Object)*: Returns the new object.

**Example**

\_.zipObject([['fred', 30], ['barney', 40]]);

// => { 'fred': 30, 'barney': 40 }

\_.zipObject(['fred', 'barney'], [30, 40]);

// => { 'fred': 30, 'barney': 40 }

**\_.zipWith([arrays], [iteratee], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_zipwitharrays-iteratee-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L5982) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.zipwith)

This method is like \_.zip except that it accepts an iteratee to specify how grouped values should be combined. Theiteratee is bound to thisArg and invoked with four arguments: (accumulator, value, index, group).

**Arguments**

1. [arrays] *(...Array)*: The arrays to process.
2. [iteratee] *(Function)*: The function to combine grouped values.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Array)*: Returns the new array of grouped elements.

**Example**

\_.zipWith([1, 2], [10, 20], [100, 200], \_.add);

// => [111, 222]

**“Chain” Methods**

**\_(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_value) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L944) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Creates a lodash object which wraps value to enable implicit chaining. Methods that operate on and return arrays, collections, and functions can be chained together. Methods that retrieve a single value or may return a primitive value will automatically end the chain returning the unwrapped value. Explicit chaining may be enabled using \_.chain. The execution of chained methods is lazy, that is, execution is deferred until \_#value is implicitly or explicitly called.   
  
Lazy evaluation allows several methods to support shortcut fusion. Shortcut fusion is an optimization strategy which merge iteratee calls; this can help to avoid the creation of intermediate data structures and greatly reduce the number of iteratee executions.   
  
Chaining is supported in custom builds as long as the \_#value method is directly or indirectly included in the build.   
  
In addition to lodash methods, wrappers have Array and String methods.   
  
The wrapper Array methods are:  
concat, join, pop, push, reverse, shift, slice, sort, splice, and unshift   
  
The wrapper String methods are:  
replace and split   
  
The wrapper methods that support shortcut fusion are:  
compact, drop, dropRight, dropRightWhile, dropWhile, filter, first, initial, last, map, pluck, reject, rest,reverse, slice, take, takeRight, takeRightWhile, takeWhile, toArray, and where   
  
The chainable wrapper methods are:  
after, ary, assign, at, before, bind, bindAll, bindKey, callback, chain, chunk, commit, compact, concat,constant, countBy, create, curry, debounce, defaults, defaultsDeep, defer, delay, difference, drop, dropRight,dropRightWhile, dropWhile, fill, filter, flatten, flattenDeep, flow, flowRight, forEach, forEachRight, forIn,forInRight, forOwn, forOwnRight, functions, groupBy, indexBy, initial, intersection, invert, invoke, keys,keysIn, map, mapKeys, mapValues, matches, matchesProperty, memoize, merge, method, methodOf, mixin, modArgs,negate, omit, once, pairs, partial, partialRight, partition, pick, plant, pluck, property, propertyOf, pull,pullAt, push, range, rearg, reject, remove, rest, restParam, reverse, set, shuffle, slice, sort, sortBy,sortByAll, sortByOrder, splice, spread, take, takeRight, takeRightWhile, takeWhile, tap, throttle, thru, times,toArray, toPlainObject, transform, union, uniq, unshift, unzip, unzipWith, values, valuesIn, where, without,wrap, xor, zip, zipObject, zipWith   
  
The wrapper methods that are **not** chainable by default are:  
add, attempt, camelCase, capitalize, ceil, clone, cloneDeep, deburr, endsWith, escape, escapeRegExp, every,find, findIndex, findKey, findLast, findLastIndex, findLastKey, findWhere, first, floor, get, gt, gte, has,identity, includes, indexOf, inRange, isArguments, isArray, isBoolean, isDate, isElement, isEmpty, isEqual,isError, isFinite isFunction, isMatch, isNative, isNaN, isNull, isNumber, isObject, isPlainObject, isRegExp,isString, isUndefined, isTypedArray, join, kebabCase, last, lastIndexOf, lt, lte, max, min, noConflict, noop,now, pad, padLeft, padRight, parseInt, pop, random, reduce, reduceRight, repeat, result, round, runInContext,shift, size, snakeCase, some, sortedIndex, sortedLastIndex, startCase, startsWith, sum, template, trim,trimLeft, trimRight, trunc, unescape, uniqueId, value, and words   
  
The wrapper method sample will return a wrapped value when n is provided, otherwise an unwrapped value is returned.

**Arguments**

1. value *(\*)*: The value to wrap in a lodash instance.

**Returns**

*(Object)*: Returns the new lodash wrapper instance.

**Example**

var wrapped = \_([1, 2, 3]);

// returns an unwrapped value

wrapped.reduce(function(total, n) {

return total + n;

});

// => 6

// returns a wrapped value

var squares = wrapped.map(function(n) {

return n \* n;

});

\_.isArray(squares);

// => false

\_.isArray(squares.value());

// => true

**\_.chain(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_chainvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6025) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Creates a lodash object that wraps value with explicit method chaining enabled.

**Arguments**

1. value *(\*)*: The value to wrap.

**Returns**

*(Object)*: Returns the new lodash wrapper instance.

**Example**

var users = [

{ 'user': 'barney', 'age': 36 },

{ 'user': 'fred', 'age': 40 },

{ 'user': 'pebbles', 'age': 1 }

];

var youngest = \_.chain(users)

.sortBy('age')

.map(function(chr) {

return chr.user + ' is ' + chr.age;

})

.first()

.value();

// => 'pebbles is 1'

**\_.tap(value, interceptor, [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_tapvalue-interceptor-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6054) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

This method invokes interceptor and returns value. The interceptor is bound to thisArg and invoked with one argument; (value). The purpose of this method is to "tap into" a method chain in order to perform operations on intermediate results within the chain.

**Arguments**

1. value *(\*)*: The value to provide to interceptor.
2. interceptor *(Function)*: The function to invoke.
3. [thisArg] *(\*)*: The this binding of interceptor.

**Returns**

*(\*)*: Returns value.

**Example**

\_([1, 2, 3])

.tap(function(array) {

array.pop();

})

.reverse()

.value();

// => [2, 1]

**\_.thru(value, interceptor, [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_thruvalue-interceptor-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6080) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

This method is like \_.tap except that it returns the result of interceptor.

**Arguments**

1. value *(\*)*: The value to provide to interceptor.
2. interceptor *(Function)*: The function to invoke.
3. [thisArg] *(\*)*: The this binding of interceptor.

**Returns**

*(\*)*: Returns the result of interceptor.

**Example**

\_(' abc ')

.chain()

.trim()

.thru(function(value) {

return [value];

})

.value();

// => ['abc']

**\_.prototype.chain()**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypechain) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6109) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Enables explicit method chaining on the wrapper object.

**Returns**

*(Object)*: Returns the new lodash wrapper instance.

**Example**

var users = [

{ 'user': 'barney', 'age': 36 },

{ 'user': 'fred', 'age': 40 }

];

// without explicit chaining

\_(users).first();

// => { 'user': 'barney', 'age': 36 }

// with explicit chaining

\_(users).chain()

.first()

.pick('user')

.value();

// => { 'user': 'barney' }

**\_.prototype.commit()**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypecommit) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6138) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Executes the chained sequence and returns the wrapped result.

**Returns**

*(Object)*: Returns the new lodash wrapper instance.

**Example**

var array = [1, 2];

var wrapped = \_(array).push(3);

console.log(array);

// => [1, 2]

wrapped = wrapped.commit();

console.log(array);

// => [1, 2, 3]

wrapped.last();

// => 3

console.log(array);

// => [1, 2, 3]

**\_.prototype.concat([values])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypeconcatvalues) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6162) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Creates a new array joining a wrapped array with any additional arrays and/or values.

**Arguments**

1. [values] *(...\*)*: The values to concatenate.

**Returns**

*(Array)*: Returns the new concatenated array.

**Example**

var array = [1];

var wrapped = \_(array).concat(2, [3], [[4]]);

console.log(wrapped.value());

// => [1, 2, 3, [4]]

console.log(array);

// => [1]

**\_.prototype.plant()**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypeplant) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6192) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Creates a clone of the chained sequence planting value as the wrapped value.

**Returns**

*(Object)*: Returns the new lodash wrapper instance.

**Example**

var array = [1, 2];

var wrapped = \_(array).map(function(value) {

return Math.pow(value, 2);

});

var other = [3, 4];

var otherWrapped = wrapped.plant(other);

otherWrapped.value();

// => [9, 16]

wrapped.value();

// => [1, 4]

**\_.prototype.reverse()**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypereverse) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6230) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Reverses the wrapped array so the first element becomes the last, the second element becomes the second to last, and so on.   
  
**Note:** This method mutates the wrapped array.

**Returns**

*(Object)*: Returns the new reversed lodash wrapper instance.

**Example**

var array = [1, 2, 3];

\_(array).reverse().value()

// => [3, 2, 1]

console.log(array);

// => [3, 2, 1]

**\_.prototype.toString()**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypetostring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6260) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Produces the result of coercing the unwrapped value to a string.

**Returns**

*(string)*: Returns the coerced string value.

**Example**

\_([1, 2, 3]).toString();

// => '1,2,3'

**\_.prototype.value()**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_prototypevalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6277) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

Executes the chained sequence to extract the unwrapped value.

**Aliases**

*\_.prototype.run, \_.prototype.toJSON, \_.prototype.valueOf*

**Returns**

*(\*)*: Returns the resolved unwrapped value.

**Example**

\_([1, 2, 3]).value();

// => [1, 2, 3]

**“Collection” Methods**

**\_.at(collection, [props])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_atcollection-props) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6303) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.at)

Creates an array of elements corresponding to the given keys, or indexes, of collection. Keys may be specified as individual arguments or as arrays of keys.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [props] *(...(number|number[]|string|string[])*: The property names or indexes of elements to pick, specified individually or in arrays.

**Returns**

*(Array)*: Returns the new array of picked elements.

**Example**

\_.at(['a', 'b', 'c'], [0, 2]);

// => ['a', 'c']

\_.at(['barney', 'fred', 'pebbles'], 0, 2);

// => ['barney', 'pebbles']

**\_.countBy(collection, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_countbycollection-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6351) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.countby)

Creates an object composed of keys generated from the results of running each element of collection through iteratee. The corresponding value of each key is the number of times the key was returned by iteratee. The iteratee is bound tothisArg and invoked with three arguments:  
(value, index|key, collection).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns the composed aggregate object.

**Example**

\_.countBy([4.3, 6.1, 6.4], function(n) {

return Math.floor(n);

});

// => { '4': 1, '6': 2 }

\_.countBy([4.3, 6.1, 6.4], function(n) {

return this.floor(n);

}, Math);

// => { '4': 1, '6': 2 }

\_.countBy(['one', 'two', 'three'], 'length');

// => { '3': 2, '5': 1 }

**\_.every(collection, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_everycollection-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6403) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.every)

Checks if predicate returns truthy for **all** elements of collection. The predicate is bound to thisArg and invoked with three arguments:  
(value, index|key, collection).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Aliases**

*\_.all*

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(boolean)*: Returns true if all elements pass the predicate check, else false.

**Example**

\_.every([true, 1, null, 'yes'], Boolean);

// => false

var users = [

{ 'user': 'barney', 'active': false },

{ 'user': 'fred', 'active': false }

];

// using the `\_.matches` callback shorthand

\_.every(users, { 'user': 'barney', 'active': false });

// => false

// using the `\_.matchesProperty` callback shorthand

\_.every(users, 'active', false);

// => true

// using the `\_.property` callback shorthand

\_.every(users, 'active');

// => false

**\_.filter(collection, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_filtercollection-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6463) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.filter)

Iterates over elements of collection, returning an array of all elements predicate returns truthy for. The predicate is bound to thisArg and invoked with three arguments: (value, index|key, collection).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Aliases**

*\_.select*

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Array)*: Returns the new filtered array.

**Example**

\_.filter([4, 5, 6], function(n) {

return n % 2 == 0;

});

// => [4, 6]

var users = [

{ 'user': 'barney', 'age': 36, 'active': true },

{ 'user': 'fred', 'age': 40, 'active': false }

];

// using the `\_.matches` callback shorthand

\_.pluck(\_.filter(users, { 'age': 36, 'active': true }), 'user');

// => ['barney']

// using the `\_.matchesProperty` callback shorthand

\_.pluck(\_.filter(users, 'active', false), 'user');

// => ['fred']

// using the `\_.property` callback shorthand

\_.pluck(\_.filter(users, 'active'), 'user');

// => ['barney']

**\_.find(collection, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_findcollection-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6519) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.find)

Iterates over elements of collection, returning the first element predicate returns truthy for. The predicate is bound tothisArg and invoked with three arguments: (value, index|key, collection).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Aliases**

*\_.detect*

**Arguments**

1. collection *(Array|Object|string)*: The collection to search.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(\*)*: Returns the matched element, else undefined.

**Example**

var users = [

{ 'user': 'barney', 'age': 36, 'active': true },

{ 'user': 'fred', 'age': 40, 'active': false },

{ 'user': 'pebbles', 'age': 1, 'active': true }

];

\_.result(\_.find(users, function(chr) {

return chr.age < 40;

}), 'user');

// => 'barney'

// using the `\_.matches` callback shorthand

\_.result(\_.find(users, { 'age': 1, 'active': true }), 'user');

// => 'pebbles'

// using the `\_.matchesProperty` callback shorthand

\_.result(\_.find(users, 'active', false), 'user');

// => 'fred'

// using the `\_.property` callback shorthand

\_.result(\_.find(users, 'active'), 'user');

// => 'barney'

**\_.findLast(collection, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_findlastcollection-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6540) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.findlast)

This method is like \_.find except that it iterates over elements of collection from right to left.

**Arguments**

1. collection *(Array|Object|string)*: The collection to search.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(\*)*: Returns the matched element, else undefined.

**Example**

\_.findLast([1, 2, 3, 4], function(n) {

return n % 2 == 1;

});

// => 3

**\_.findWhere(collection, source)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_findwherecollection-source) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6571) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.findwhere)

Performs a deep comparison between each element in collection and the source object, returning the first element that has equivalent property values.   
  
**Note:** This method supports comparing arrays, booleans, Date objects, numbers, Object objects, regexes, and strings. Objects are compared by their own, not inherited, enumerable properties. For comparing a single own or inherited property value see \_.matchesProperty.

**Arguments**

1. collection *(Array|Object|string)*: The collection to search.
2. source *(Object)*: The object of property values to match.

**Returns**

*(\*)*: Returns the matched element, else undefined.

**Example**

var users = [

{ 'user': 'barney', 'age': 36, 'active': true },

{ 'user': 'fred', 'age': 40, 'active': false }

];

\_.result(\_.findWhere(users, { 'age': 36, 'active': true }), 'user');

// => 'barney'

\_.result(\_.findWhere(users, { 'age': 40, 'active': false }), 'user');

// => 'fred'

**\_.forEach(collection, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_foreachcollection-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6605) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.foreach)

Iterates over elements of collection invoking iteratee for each element. The iteratee is bound to thisArg and invoked with three arguments:  
(value, index|key, collection). Iteratee functions may exit iteration early by explicitly returning false.   
  
**Note:** As with other "Collections" methods, objects with a "length" property are iterated like arrays. To avoid this behavior\_.forIn or \_.forOwn may be used for object iteration.

**Aliases**

*\_.each*

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Array|Object|string)*: Returns collection.

**Example**

\_([1, 2]).forEach(function(n) {

console.log(n);

}).value();

// => logs each value from left to right and returns the array

\_.forEach({ 'a': 1, 'b': 2 }, function(n, key) {

console.log(n, key);

});

// => logs each value-key pair and returns the object (iteration order is not guaranteed)

**\_.forEachRight(collection, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_foreachrightcollection-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6626) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.foreachright)

This method is like \_.forEach except that it iterates over elements of collection from right to left.

**Aliases**

*\_.eachRight*

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Array|Object|string)*: Returns collection.

**Example**

\_([1, 2]).forEachRight(function(n) {

console.log(n);

}).value();

// => logs each value from right to left and returns the array

**\_.groupBy(collection, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_groupbycollection-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6670) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.groupby)

Creates an object composed of keys generated from the results of running each element of collection through iteratee. The corresponding value of each key is an array of the elements responsible for generating the key. The iteratee is bound to thisArg and invoked with three arguments:  
(value, index|key, collection).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns the composed aggregate object.

**Example**

\_.groupBy([4.2, 6.1, 6.4], function(n) {

return Math.floor(n);

});

// => { '4': [4.2], '6': [6.1, 6.4] }

\_.groupBy([4.2, 6.1, 6.4], function(n) {

return this.floor(n);

}, Math);

// => { '4': [4.2], '6': [6.1, 6.4] }

// using the `\_.property` callback shorthand

\_.groupBy(['one', 'two', 'three'], 'length');

// => { '3': ['one', 'two'], '5': ['three'] }

**\_.includes(collection, target, [fromIndex=0])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_includescollection-target-fromindex0) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6707) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.includes)

Checks if target is in collection using [SameValueZero](http://ecma-international.org/ecma-262/6.0/#sec-samevaluezero) for equality comparisons. If fromIndex is negative, it's used as the offset from the end of collection.

**Aliases**

*\_.contains, \_.include*

**Arguments**

1. collection *(Array|Object|string)*: The collection to search.
2. target *(\*)*: The value to search for.
3. [fromIndex=0] *(number)*: The index to search from.

**Returns**

*(boolean)*: Returns true if a matching element is found, else false.

**Example**

\_.includes([1, 2, 3], 1);

// => true

\_.includes([1, 2, 3], 1, 2);

// => false

\_.includes({ 'user': 'fred', 'age': 40 }, 'fred');

// => true

\_.includes('pebbles', 'eb');

// => true

**\_.indexBy(collection, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_indexbycollection-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6769) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.indexby)

Creates an object composed of keys generated from the results of running each element of collection through iteratee. The corresponding value of each key is the last element responsible for generating the key. The iteratee function is bound tothisArg and invoked with three arguments:  
(value, index|key, collection).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns the composed aggregate object.

**Example**

var keyData = [

{ 'dir': 'left', 'code': 97 },

{ 'dir': 'right', 'code': 100 }

];

\_.indexBy(keyData, 'dir');

// => { 'left': { 'dir': 'left', 'code': 97 }, 'right': { 'dir': 'right', 'code': 100 } }

\_.indexBy(keyData, function(object) {

return String.fromCharCode(object.code);

});

// => { 'a': { 'dir': 'left', 'code': 97 }, 'd': { 'dir': 'right', 'code': 100 } }

\_.indexBy(keyData, function(object) {

return this.fromCharCode(object.code);

}, String);

// => { 'a': { 'dir': 'left', 'code': 97 }, 'd': { 'dir': 'right', 'code': 100 } }

**\_.invoke(collection, path, [args])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_invokecollection-path-args) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6795) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.invoke)

Invokes the method at path of each element in collection, returning an array of the results of each invoked method. Any additional arguments are provided to each invoked method. If methodName is a function it's invoked for, and this bound to, each element in collection.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. path *(Array|Function|string)*: The path of the method to invoke or the function invoked per iteration.
3. [args] *(...\*)*: The arguments to invoke the method with.

**Returns**

*(Array)*: Returns the array of results.

**Example**

\_.invoke([[5, 1, 7], [3, 2, 1]], 'sort');

// => [[1, 5, 7], [1, 2, 3]]

\_.invoke([123, 456], String.prototype.split, '');

// => [['1', '2', '3'], ['4', '5', '6']]

**\_.map(collection, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_mapcollection-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6864) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.map)

Creates an array of values by running each element in collection through iteratee. The iteratee is bound to thisArgand invoked with three arguments: (value, index|key, collection).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.   
  
Many lodash methods are guarded to work as iteratees for methods like \_.every, \_.filter, \_.map, \_.mapValues,\_.reject, and \_.some.   
  
The guarded methods are:  
ary, callback, chunk, clone, create, curry, curryRight, drop, dropRight, every, fill, flatten, invert, max,min, parseInt, slice, sortBy, take, takeRight, template, trim, trimLeft, trimRight, trunc, random, range,sample, some, sum, uniq, and words

**Aliases**

*\_.collect*

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Array)*: Returns the new mapped array.

**Example**

function timesThree(n) {

return n \* 3;

}

\_.map([1, 2], timesThree);

// => [3, 6]

\_.map({ 'a': 1, 'b': 2 }, timesThree);

// => [3, 6] (iteration order is not guaranteed)

var users = [

{ 'user': 'barney' },

{ 'user': 'fred' }

];

// using the `\_.property` callback shorthand

\_.map(users, 'user');

// => ['barney', 'fred']

**\_.partition(collection, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_partitioncollection-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6929) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.partition)

Creates an array of elements split into two groups, the first of which contains elements predicate returns truthy for, while the second of which contains elements predicate returns falsey for. The predicate is bound to thisArg and invoked with three arguments: (value, index|key, collection).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Array)*: Returns the array of grouped elements.

**Example**

\_.partition([1, 2, 3], function(n) {

return n % 2;

});

// => [[1, 3], [2]]

\_.partition([1.2, 2.3, 3.4], function(n) {

return this.floor(n) % 2;

}, Math);

// => [[1.2, 3.4], [2.3]]

var users = [

{ 'user': 'barney', 'age': 36, 'active': false },

{ 'user': 'fred', 'age': 40, 'active': true },

{ 'user': 'pebbles', 'age': 1, 'active': false }

];

var mapper = function(array) {

return \_.pluck(array, 'user');

};

// using the `\_.matches` callback shorthand

\_.map(\_.partition(users, { 'age': 1, 'active': false }), mapper);

// => [['pebbles'], ['barney', 'fred']]

// using the `\_.matchesProperty` callback shorthand

\_.map(\_.partition(users, 'active', false), mapper);

// => [['barney', 'pebbles'], ['fred']]

// using the `\_.property` callback shorthand

\_.map(\_.partition(users, 'active'), mapper);

// => [['fred'], ['barney', 'pebbles']]

**\_.pluck(collection, path)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_pluckcollection-path) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6956) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.pluck)

Gets the property value of path from all elements in collection.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. path *(Array|string)*: The path of the property to pluck.

**Returns**

*(Array)*: Returns the property values.

**Example**

var users = [

{ 'user': 'barney', 'age': 36 },

{ 'user': 'fred', 'age': 40 }

];

\_.pluck(users, 'user');

// => ['barney', 'fred']

var userIndex = \_.indexBy(users, 'user');

\_.pluck(userIndex, 'age');

// => [36, 40] (iteration order is not guaranteed)

**\_.reduce(collection, [iteratee=\_.identity], [accumulator], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_reducecollection-iteratee_identity-accumulator-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L6997) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.reduce)

Reduces collection to a value which is the accumulated result of running each element in collection through iteratee, where each successive invocation is supplied the return value of the previous. If accumulator is not provided the first element of collection is used as the initial value. The iteratee is bound to thisArg and invoked with four arguments:  
(accumulator, value, index|key, collection).   
  
Many lodash methods are guarded to work as iteratees for methods like \_.reduce, \_.reduceRight, and \_.transform.   
  
The guarded methods are:  
assign, defaults, defaultsDeep, includes, merge, sortByAll, and sortByOrder

**Aliases**

*\_.foldl, \_.inject*

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [accumulator] *(\*)*: The initial value.
4. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(\*)*: Returns the accumulated value.

**Example**

\_.reduce([1, 2], function(total, n) {

return total + n;

});

// => 3

\_.reduce({ 'a': 1, 'b': 2 }, function(result, n, key) {

result[key] = n \* 3;

return result;

}, {});

// => { 'a': 3, 'b': 6 } (iteration order is not guaranteed)

**\_.reduceRight(collection, [iteratee=\_.identity], [accumulator], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_reducerightcollection-iteratee_identity-accumulator-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7021) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.reduceright)

This method is like \_.reduce except that it iterates over elements of collection from right to left.

**Aliases**

*\_.foldr*

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [accumulator] *(\*)*: The initial value.
4. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(\*)*: Returns the accumulated value.

**Example**

var array = [[0, 1], [2, 3], [4, 5]];

\_.reduceRight(array, function(flattened, other) {

return flattened.concat(other);

}, []);

// => [4, 5, 2, 3, 0, 1]

**\_.reject(collection, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_rejectcollection-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7059) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.reject)

The opposite of \_.filter; this method returns the elements of collection that predicate does **not** return truthy for.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Array)*: Returns the new filtered array.

**Example**

\_.reject([1, 2, 3, 4], function(n) {

return n % 2 == 0;

});

// => [1, 3]

var users = [

{ 'user': 'barney', 'age': 36, 'active': false },

{ 'user': 'fred', 'age': 40, 'active': true }

];

// using the `\_.matches` callback shorthand

\_.pluck(\_.reject(users, { 'age': 40, 'active': true }), 'user');

// => ['barney']

// using the `\_.matchesProperty` callback shorthand

\_.pluck(\_.reject(users, 'active', false), 'user');

// => ['fred']

// using the `\_.property` callback shorthand

\_.pluck(\_.reject(users, 'active'), 'user');

// => ['barney']

**\_.sample(collection, [n])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_samplecollection-n) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7085) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.sample)

Gets a random element or n random elements from a collection.

**Arguments**

1. collection *(Array|Object|string)*: The collection to sample.
2. [n] *(number)*: The number of elements to sample.

**Returns**

*(\*)*: Returns the random sample(s).

**Example**

\_.sample([1, 2, 3, 4]);

// => 2

\_.sample([1, 2, 3, 4], 2);

// => [3, 1]

**\_.shuffle(collection)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_shufflecollection) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7122) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.shuffle)

Creates an array of shuffled values, using a version of the [Fisher-Yates shuffle](https://en.wikipedia.org/wiki/Fisher-Yates_shuffle).

**Arguments**

1. collection *(Array|Object|string)*: The collection to shuffle.

**Returns**

*(Array)*: Returns the new shuffled array.

**Example**

\_.shuffle([1, 2, 3, 4]);

// => [4, 1, 3, 2]

**\_.size(collection)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_sizecollection) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7146) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.size)

Gets the size of collection by returning its length for array-like values or the number of own enumerable properties for objects.

**Arguments**

1. collection *(Array|Object|string)*: The collection to inspect.

**Returns**

*(number)*: Returns the size of collection.

**Example**

\_.size([1, 2, 3]);

// => 3

\_.size({ 'a': 1, 'b': 2 });

// => 2

\_.size('pebbles');

// => 7

**\_.some(collection, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_somecollection-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7200) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.some)

Checks if predicate returns truthy for **any** element of collection. The function returns as soon as it finds a passing value and does not iterate over the entire collection. The predicate is bound to thisArg and invoked with three arguments: (value, index|key, collection).   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Aliases**

*\_.any*

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(boolean)*: Returns true if any element passes the predicate check, else false.

**Example**

\_.some([null, 0, 'yes', false], Boolean);

// => true

var users = [

{ 'user': 'barney', 'active': true },

{ 'user': 'fred', 'active': false }

];

// using the `\_.matches` callback shorthand

\_.some(users, { 'user': 'barney', 'active': false });

// => false

// using the `\_.matchesProperty` callback shorthand

\_.some(users, 'active', false);

// => true

// using the `\_.property` callback shorthand

\_.some(users, 'active');

// => true

**\_.sortBy(collection, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortbycollection-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7259) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.sortby)

Creates an array of elements, sorted in ascending order by the results of running each element in a collection throughiteratee. This method performs a stable sort, that is, it preserves the original sort order of equal elements. The iteratee is bound to thisArg and invoked with three arguments:  
(value, index|key, collection).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Array)*: Returns the new sorted array.

**Example**

\_.sortBy([1, 2, 3], function(n) {

return Math.sin(n);

});

// => [3, 1, 2]

\_.sortBy([1, 2, 3], function(n) {

return this.sin(n);

}, Math);

// => [3, 1, 2]

var users = [

{ 'user': 'fred' },

{ 'user': 'pebbles' },

{ 'user': 'barney' }

];

// using the `\_.property` callback shorthand

\_.pluck(\_.sortBy(users, 'user'), 'user');

// => ['barney', 'fred', 'pebbles']

**\_.sortByAll(collection, iteratees)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortbyallcollection-iteratees) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7310) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.sortbyall)

This method is like \_.sortBy except that it can sort by multiple iteratees or property names.   
  
If a property name is provided for an iteratee the created \_.property style callback returns the property value of the given element.   
  
If an object is provided for an iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. iteratees *(...(Function|Function[]|Object|Object[]|string|string[])*: The iteratees to sort by, specified as individual values or arrays of values.

**Returns**

*(Array)*: Returns the new sorted array.

**Example**

var users = [

{ 'user': 'fred', 'age': 48 },

{ 'user': 'barney', 'age': 36 },

{ 'user': 'fred', 'age': 42 },

{ 'user': 'barney', 'age': 34 }

];

\_.map(\_.sortByAll(users, ['user', 'age']), \_.values);

// => [['barney', 34], ['barney', 36], ['fred', 42], ['fred', 48]]

\_.map(\_.sortByAll(users, 'user', function(chr) {

return Math.floor(chr.age / 10);

}), \_.values);

// => [['barney', 36], ['barney', 34], ['fred', 48], ['fred', 42]]

**\_.sortByOrder(collection, iteratees, [orders])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_sortbyordercollection-iteratees-orders) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7355) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.sortbyorder)

This method is like \_.sortByAll except that it allows specifying the sort orders of the iteratees to sort by. If orders is unspecified, all values are sorted in ascending order. Otherwise, a value is sorted in ascending order if its corresponding order is "asc", and descending if "desc".   
  
If a property name is provided for an iteratee the created \_.property style callback returns the property value of the given element.   
  
If an object is provided for an iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. iteratees *(Function[]|Object[]|string[])*: The iteratees to sort by.
3. [orders] *(boolean[])*: The sort orders of iteratees.

**Returns**

*(Array)*: Returns the new sorted array.

**Example**

var users = [

{ 'user': 'fred', 'age': 48 },

{ 'user': 'barney', 'age': 34 },

{ 'user': 'fred', 'age': 42 },

{ 'user': 'barney', 'age': 36 }

];

// sort by `user` in ascending order and by `age` in descending order

\_.map(\_.sortByOrder(users, ['user', 'age'], ['asc', 'desc']), \_.values);

// => [['barney', 36], ['barney', 34], ['fred', 48], ['fred', 42]]

**\_.where(collection, source)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_wherecollection-source) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7400) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.where)

Performs a deep comparison between each element in collection and the source object, returning an array of all elements that have equivalent property values.   
  
**Note:** This method supports comparing arrays, booleans, Date objects, numbers, Object objects, regexes, and strings. Objects are compared by their own, not inherited, enumerable properties. For comparing a single own or inherited property value see \_.matchesProperty.

**Arguments**

1. collection *(Array|Object|string)*: The collection to search.
2. source *(Object)*: The object of property values to match.

**Returns**

*(Array)*: Returns the new filtered array.

**Example**

var users = [

{ 'user': 'barney', 'age': 36, 'active': false, 'pets': ['hoppy'] },

{ 'user': 'fred', 'age': 40, 'active': true, 'pets': ['baby puss', 'dino'] }

];

\_.pluck(\_.where(users, { 'age': 36, 'active': false }), 'user');

// => ['barney']

\_.pluck(\_.where(users, { 'pets': ['dino'] }), 'user');

// => ['fred']

**“Date” Methods**

**\_.now**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_now) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7420) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.now)

Gets the number of milliseconds that have elapsed since the Unix epoch (1 January 1970 00:00:00 UTC).

**Example**

\_.defer(function(stamp) {

console.log(\_.now() - stamp);

}, \_.now());

// => logs the number of milliseconds it took for the deferred function to be invoked

**“Function” Methods**

**\_.after(n, func)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_aftern-func) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7449) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.after)

The opposite of \_.before; this method creates a function that invokes func once it's called n or more times.

**Arguments**

1. n *(number)*: The number of calls before func is invoked.
2. func *(Function)*: The function to restrict.

**Returns**

*(Function)*: Returns the new restricted function.

**Example**

var saves = ['profile', 'settings'];

var done = \_.after(saves.length, function() {

console.log('done saving!');

});

\_.forEach(saves, function(type) {

asyncSave({ 'type': type, 'complete': done });

});

// => logs 'done saving!' after the two async saves have completed

**\_.ary(func, [n=func.length])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_aryfunc-nfunclength) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7483) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.ary)

Creates a function that accepts up to n arguments ignoring any additional arguments.

**Arguments**

1. func *(Function)*: The function to cap arguments for.
2. [n=func.length] *(number)*: The arity cap.

**Returns**

*(Function)*: Returns the new function.

**Example**

\_.map(['6', '8', '10'], \_.ary(parseInt, 1));

// => [6, 8, 10]

**\_.before(n, func)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_beforen-func) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7507) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.before)

Creates a function that invokes func, with the this binding and arguments of the created function, while it's called less thann times. Subsequent calls to the created function return the result of the last func invocation.

**Arguments**

1. n *(number)*: The number of calls at which func is no longer invoked.
2. func *(Function)*: The function to restrict.

**Returns**

*(Function)*: Returns the new restricted function.

**Example**

jQuery('#add').on('click', \_.before(5, addContactToList));

// => allows adding up to 4 contacts to the list

**\_.bind(func, thisArg, [partials])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_bindfunc-thisarg-partials) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7564) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.bind)

Creates a function that invokes func with the this binding of thisArg and prepends any additional \_.bind arguments to those provided to the bound function.   
  
The \_.bind.placeholder value, which defaults to \_ in monolithic builds, may be used as a placeholder for partially applied arguments.   
  
**Note:** Unlike native Function#bind this method does not set the "length" property of bound functions.

**Arguments**

1. func *(Function)*: The function to bind.
2. thisArg *(\*)*: The this binding of func.
3. [partials] *(...\*)*: The arguments to be partially applied.

**Returns**

*(Function)*: Returns the new bound function.

**Example**

var greet = function(greeting, punctuation) {

return greeting + ' ' + this.user + punctuation;

};

var object = { 'user': 'fred' };

var bound = \_.bind(greet, object, 'hi');

bound('!');

// => 'hi fred!'

// using placeholders

var bound = \_.bind(greet, object, \_, '!');

bound('hi');

// => 'hi fred!'

**\_.bindAll(object, [methodNames])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_bindallobject-methodnames) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7601) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.bindall)

Binds methods of an object to the object itself, overwriting the existing method. Method names may be specified as individual arguments or as arrays of method names. If no method names are provided all enumerable function properties, own and inherited, of object are bound.   
  
**Note:** This method does not set the "length" property of bound functions.

**Arguments**

1. object *(Object)*: The object to bind and assign the bound methods to.
2. [methodNames] *(...(string|string[])*: The object method names to bind, specified as individual method names or arrays of method names.

**Returns**

*(Object)*: Returns object.

**Example**

var view = {

'label': 'docs',

'onClick': function() {

console.log('clicked ' + this.label);

}

};

\_.bindAll(view);

jQuery('#docs').on('click', view.onClick);

// => logs 'clicked docs' when the element is clicked

**\_.bindKey(object, key, [partials])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_bindkeyobject-key-partials) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7658) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.bindkey)

Creates a function that invokes the method at object[key] and prepends any additional \_.bindKey arguments to those provided to the bound function.   
  
This method differs from \_.bind by allowing bound functions to reference methods that may be redefined or don't yet exist. See [Peter Michaux's article](http://peter.michaux.ca/articles/lazy-function-definition-pattern) for more details.   
  
The \_.bindKey.placeholder value, which defaults to \_ in monolithic builds, may be used as a placeholder for partially applied arguments.

**Arguments**

1. object *(Object)*: The object the method belongs to.
2. key *(string)*: The key of the method.
3. [partials] *(...\*)*: The arguments to be partially applied.

**Returns**

*(Function)*: Returns the new bound function.

**Example**

var object = {

'user': 'fred',

'greet': function(greeting, punctuation) {

return greeting + ' ' + this.user + punctuation;

}

};

var bound = \_.bindKey(object, 'greet', 'hi');

bound('!');

// => 'hi fred!'

object.greet = function(greeting, punctuation) {

return greeting + 'ya ' + this.user + punctuation;

};

bound('!');

// => 'hiya fred!'

// using placeholders

var bound = \_.bindKey(object, 'greet', \_, '!');

bound('hi');

// => 'hiya fred!'

**\_.curry(func, [arity=func.length])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_curryfunc-arityfunclength) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7707) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.curry)

Creates a function that accepts one or more arguments of func that when called either invokes func returning its result, if all func arguments have been provided, or returns a function that accepts one or more of the remaining func arguments, and so on. The arity of func may be specified if func.length is not sufficient.   
  
The \_.curry.placeholder value, which defaults to \_ in monolithic builds, may be used as a placeholder for provided arguments.   
  
**Note:** This method does not set the "length" property of curried functions.

**Arguments**

1. func *(Function)*: The function to curry.
2. [arity=func.length] *(number)*: The arity of func.

**Returns**

*(Function)*: Returns the new curried function.

**Example**

var abc = function(a, b, c) {

return [a, b, c];

};

var curried = \_.curry(abc);

curried(1)(2)(3);

// => [1, 2, 3]

curried(1, 2)(3);

// => [1, 2, 3]

curried(1, 2, 3);

// => [1, 2, 3]

// using placeholders

curried(1)(\_, 3)(2);

// => [1, 2, 3]

**\_.curryRight(func, [arity=func.length])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_curryrightfunc-arityfunclength) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7746) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.curryright)

This method is like \_.curry except that arguments are applied to func in the manner of \_.partialRight instead of\_.partial.   
  
The \_.curryRight.placeholder value, which defaults to \_ in monolithic builds, may be used as a placeholder for provided arguments.   
  
**Note:** This method does not set the "length" property of curried functions.

**Arguments**

1. func *(Function)*: The function to curry.
2. [arity=func.length] *(number)*: The arity of func.

**Returns**

*(Function)*: Returns the new curried function.

**Example**

var abc = function(a, b, c) {

return [a, b, c];

};

var curried = \_.curryRight(abc);

curried(3)(2)(1);

// => [1, 2, 3]

curried(2, 3)(1);

// => [1, 2, 3]

curried(1, 2, 3);

// => [1, 2, 3]

// using placeholders

curried(3)(1, \_)(2);

// => [1, 2, 3]

**\_.debounce(func, [wait=0], [options])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_debouncefunc-wait0-options) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7811) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.debounce)

Creates a debounced function that delays invoking func until after wait milliseconds have elapsed since the last time the debounced function was invoked. The debounced function comes with a cancel method to cancel delayed invocations. Provide an options object to indicate that func should be invoked on the leading and/or trailing edge of the wait timeout. Subsequent calls to the debounced function return the result of the last func invocation.   
  
**Note:** If leading and trailing options are true, func is invoked on the trailing edge of the timeout only if the the debounced function is invoked more than once during the wait timeout.   
  
See [David Corbacho's article](http://drupalmotion.com/article/debounce-and-throttle-visual-explanation) for details over the differences between \_.debounce and \_.throttle.

**Arguments**

1. func *(Function)*: The function to debounce.
2. [wait=0] *(number)*: The number of milliseconds to delay.
3. [options] *(Object)*: The options object.
4. [options.leading=false] *(boolean)*: Specify invoking on the leading edge of the timeout.
5. [options.maxWait] *(number)*: The maximum time func is allowed to be delayed before it's invoked.
6. [options.trailing=true] *(boolean)*: Specify invoking on the trailing edge of the timeout.

**Returns**

*(Function)*: Returns the new debounced function.

**Example**

// avoid costly calculations while the window size is in flux

jQuery(window).on('resize', \_.debounce(calculateLayout, 150));

// invoke `sendMail` when the click event is fired, debouncing subsequent calls

jQuery('#postbox').on('click', \_.debounce(sendMail, 300, {

'leading': true,

'trailing': false

}));

// ensure `batchLog` is invoked once after 1 second of debounced calls

var source = new EventSource('/stream');

jQuery(source).on('message', \_.debounce(batchLog, 250, {

'maxWait': 1000

}));

// cancel a debounced call

var todoChanges = \_.debounce(batchLog, 1000);

Object.observe(models.todo, todoChanges);

Object.observe(models, function(changes) {

if (\_.find(changes, { 'user': 'todo', 'type': 'delete'})) {

todoChanges.cancel();

}

}, ['delete']);

// ...at some point `models.todo` is changed

models.todo.completed = true;

// ...before 1 second has passed `models.todo` is deleted

// which cancels the debounced `todoChanges` call

delete models.todo;

**\_.defer(func, [args])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_deferfunc-args) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7936) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.defer)

Defers invoking the func until the current call stack has cleared. Any additional arguments are provided to func when it's invoked.

**Arguments**

1. func *(Function)*: The function to defer.
2. [args] *(...\*)*: The arguments to invoke the function with.

**Returns**

*(number)*: Returns the timer id.

**Example**

\_.defer(function(text) {

console.log(text);

}, 'deferred');

// logs 'deferred' after one or more milliseconds

**\_.delay(func, wait, [args])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_delayfunc-wait-args) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7958) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.delay)

Invokes func after wait milliseconds. Any additional arguments are provided to func when it's invoked.

**Arguments**

1. func *(Function)*: The function to delay.
2. wait *(number)*: The number of milliseconds to delay invocation.
3. [args] *(...\*)*: The arguments to invoke the function with.

**Returns**

*(number)*: Returns the timer id.

**Example**

\_.delay(function(text) {

console.log(text);

}, 1000, 'later');

// => logs 'later' after one second

**\_.flow([funcs])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_flowfuncs) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L7982) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.flow)

Creates a function that returns the result of invoking the provided functions with the this binding of the created function, where each successive invocation is supplied the return value of the previous.

**Arguments**

1. [funcs] *(...Function)*: Functions to invoke.

**Returns**

*(Function)*: Returns the new function.

**Example**

function square(n) {

return n \* n;

}

var addSquare = \_.flow(\_.add, square);

addSquare(1, 2);

// => 9

**\_.flowRight([funcs])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_flowrightfuncs) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8004) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.flowright)

This method is like \_.flow except that it creates a function that invokes the provided functions from right to left.

**Aliases**

*\_.backflow, \_.compose*

**Arguments**

1. [funcs] *(...Function)*: Functions to invoke.

**Returns**

*(Function)*: Returns the new function.

**Example**

function square(n) {

return n \* n;

}

var addSquare = \_.flowRight(square, \_.add);

addSquare(1, 2);

// => 9

**\_.memoize(func, [resolver])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_memoizefunc-resolver) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8057) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.memoize)

Creates a function that memoizes the result of func. If resolver is provided it determines the cache key for storing the result based on the arguments provided to the memoized function. By default, the first argument provided to the memoized function is coerced to a string and used as the cache key. The func is invoked with the this binding of the memoized function.   
  
**Note:** The cache is exposed as the cache property on the memoized function. Its creation may be customized by replacing the \_.memoize.Cache constructor with one whose instances implement the [Map](http://ecma-international.org/ecma-262/6.0/#sec-properties-of-the-map-prototype-object) method interface of get, has, and set.

**Arguments**

1. func *(Function)*: The function to have its output memoized.
2. [resolver] *(Function)*: The function to resolve the cache key.

**Returns**

*(Function)*: Returns the new memoizing function.

**Example**

var upperCase = \_.memoize(function(string) {

return string.toUpperCase();

});

upperCase('fred');

// => 'FRED'

// modifying the result cache

upperCase.cache.set('fred', 'BARNEY');

upperCase('fred');

// => 'BARNEY'

// replacing `\_.memoize.Cache`

var object = { 'user': 'fred' };

var other = { 'user': 'barney' };

var identity = \_.memoize(\_.identity);

identity(object);

// => { 'user': 'fred' }

identity(other);

// => { 'user': 'fred' }

\_.memoize.Cache = WeakMap;

var identity = \_.memoize(\_.identity);

identity(object);

// => { 'user': 'fred' }

identity(other);

// => { 'user': 'barney' }

**\_.modArgs(func, [transforms])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_modargsfunc-transforms) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8108) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.modargs)

Creates a function that runs each argument through a corresponding transform function.

**Arguments**

1. func *(Function)*: The function to wrap.
2. [transforms] *(...(Function|Function[])*: The functions to transform arguments, specified as individual functions or arrays of functions.

**Returns**

*(Function)*: Returns the new function.

**Example**

function doubled(n) {

return n \* 2;

}

function square(n) {

return n \* n;

}

var modded = \_.modArgs(function(x, y) {

return [x, y];

}, square, doubled);

modded(1, 2);

// => [1, 4]

modded(5, 10);

// => [25, 20]

**\_.negate(predicate)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_negatepredicate) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8142) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.negate)

Creates a function that negates the result of the predicate func. The func predicate is invoked with the this binding and arguments of the created function.

**Arguments**

1. predicate *(Function)*: The predicate to negate.

**Returns**

*(Function)*: Returns the new function.

**Example**

function isEven(n) {

return n % 2 == 0;

}

\_.filter([1, 2, 3, 4, 5, 6], \_.negate(isEven));

// => [1, 3, 5]

**\_.once(func)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_oncefunc) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8168) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.once)

Creates a function that is restricted to invoking func once. Repeat calls to the function return the value of the first call. Thefunc is invoked with the this binding and arguments of the created function.

**Arguments**

1. func *(Function)*: The function to restrict.

**Returns**

*(Function)*: Returns the new restricted function.

**Example**

var initialize = \_.once(createApplication);

initialize();

initialize();

// `initialize` invokes `createApplication` once

**\_.partial(func, [partials])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_partialfunc-partials) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8204) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.partial)

Creates a function that invokes func with partial arguments prepended to those provided to the new function. This method is like \_.bind except it does **not** alter the this binding.   
  
The \_.partial.placeholder value, which defaults to \_ in monolithic builds, may be used as a placeholder for partially applied arguments.   
  
**Note:** This method does not set the "length" property of partially applied functions.

**Arguments**

1. func *(Function)*: The function to partially apply arguments to.
2. [partials] *(...\*)*: The arguments to be partially applied.

**Returns**

*(Function)*: Returns the new partially applied function.

**Example**

var greet = function(greeting, name) {

return greeting + ' ' + name;

};

var sayHelloTo = \_.partial(greet, 'hello');

sayHelloTo('fred');

// => 'hello fred'

// using placeholders

var greetFred = \_.partial(greet, \_, 'fred');

greetFred('hi');

// => 'hi fred'

**\_.partialRight(func, [partials])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_partialrightfunc-partials) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8237) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.partialright)

This method is like \_.partial except that partially applied arguments are appended to those provided to the new function.   
  
The \_.partialRight.placeholder value, which defaults to \_ in monolithic builds, may be used as a placeholder for partially applied arguments.   
  
**Note:** This method does not set the "length" property of partially applied functions.

**Arguments**

1. func *(Function)*: The function to partially apply arguments to.
2. [partials] *(...\*)*: The arguments to be partially applied.

**Returns**

*(Function)*: Returns the new partially applied function.

**Example**

var greet = function(greeting, name) {

return greeting + ' ' + name;

};

var greetFred = \_.partialRight(greet, 'fred');

greetFred('hi');

// => 'hi fred'

// using placeholders

var sayHelloTo = \_.partialRight(greet, 'hello', \_);

sayHelloTo('fred');

// => 'hello fred'

**\_.rearg(func, indexes)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_reargfunc-indexes) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8267) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.rearg)

Creates a function that invokes func with arguments arranged according to the specified indexes where the argument value at the first index is provided as the first argument, the argument value at the second index is provided as the second argument, and so on.

**Arguments**

1. func *(Function)*: The function to rearrange arguments for.
2. indexes *(...(number|number[])*: The arranged argument indexes, specified as individual indexes or arrays of indexes.

**Returns**

*(Function)*: Returns the new function.

**Example**

var rearged = \_.rearg(function(a, b, c) {

return [a, b, c];

}, 2, 0, 1);

rearged('b', 'c', 'a')

// => ['a', 'b', 'c']

var map = \_.rearg(\_.map, [1, 0]);

map(function(n) {

return n \* 3;

}, [1, 2, 3]);

// => [3, 6, 9]

**\_.restParam(func, [start=func.length-1])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_restparamfunc-startfunclength-1) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8293) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.restparam)

Creates a function that invokes func with the this binding of the created function and arguments from start and beyond provided as an array.   
  
**Note:** This method is based on the [rest parameter](https://developer.mozilla.org/Web/JavaScript/Reference/Functions/rest_parameters).

**Arguments**

1. func *(Function)*: The function to apply a rest parameter to.
2. [start=func.length-1] *(number)*: The start position of the rest parameter.

**Returns**

*(Function)*: Returns the new function.

**Example**

var say = \_.restParam(function(what, names) {

return what + ' ' + \_.initial(names).join(', ') +

(\_.size(names) > 1 ? ', & ' : '') + \_.last(names);

});

say('hello', 'fred', 'barney', 'pebbles');

// => 'hello fred, barney, & pebbles'

**\_.spread(func)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_spreadfunc) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8353) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.spread)

Creates a function that invokes func with the this binding of the created function and an array of arguments much like[Function#apply](https://es5.github.io/#x15.3.4.3).   
  
**Note:** This method is based on the [spread operator](https://developer.mozilla.org/Web/JavaScript/Reference/Operators/Spread_operator).

**Arguments**

1. func *(Function)*: The function to spread arguments over.

**Returns**

*(Function)*: Returns the new function.

**Example**

var say = \_.spread(function(who, what) {

return who + ' says ' + what;

});

say(['fred', 'hello']);

// => 'fred says hello'

// with a Promise

var numbers = Promise.all([

Promise.resolve(40),

Promise.resolve(36)

]);

numbers.then(\_.spread(function(x, y) {

return x + y;

}));

// => a Promise of 76

**\_.throttle(func, [wait=0], [options])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_throttlefunc-wait0-options) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8401) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.throttle)

Creates a throttled function that only invokes func at most once per every wait milliseconds. The throttled function comes with a cancel method to cancel delayed invocations. Provide an options object to indicate that func should be invoked on the leading and/or trailing edge of the wait timeout. Subsequent calls to the throttled function return the result of the lastfunc call.   
  
**Note:** If leading and trailing options are true, func is invoked on the trailing edge of the timeout only if the the throttled function is invoked more than once during the wait timeout.   
  
See [David Corbacho's article](http://drupalmotion.com/article/debounce-and-throttle-visual-explanation) for details over the differences between \_.throttle and \_.debounce.

**Arguments**

1. func *(Function)*: The function to throttle.
2. [wait=0] *(number)*: The number of milliseconds to throttle invocations to.
3. [options] *(Object)*: The options object.
4. [options.leading=true] *(boolean)*: Specify invoking on the leading edge of the timeout.
5. [options.trailing=true] *(boolean)*: Specify invoking on the trailing edge of the timeout.

**Returns**

*(Function)*: Returns the new throttled function.

**Example**

// avoid excessively updating the position while scrolling

jQuery(window).on('scroll', \_.throttle(updatePosition, 100));

// invoke `renewToken` when the click event is fired, but not more than once every 5 minutes

jQuery('.interactive').on('click', \_.throttle(renewToken, 300000, {

'trailing': false

}));

// cancel a trailing throttled call

jQuery(window).on('popstate', throttled.cancel);

**\_.wrap(value, wrapper)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_wrapvalue-wrapper) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8438) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.wrap)

Creates a function that provides value to the wrapper function as its first argument. Any additional arguments provided to the function are appended to those provided to the wrapper function. The wrapper is invoked with the this binding of the created function.

**Arguments**

1. value *(\*)*: The value to wrap.
2. wrapper *(Function)*: The wrapper function.

**Returns**

*(Function)*: Returns the new function.

**Example**

var p = \_.wrap(\_.escape, function(func, text) {

return '<p>' + func(text) + '</p>';

});

p('fred, barney, & pebbles');

// => '<p>fred, barney, &amp; pebbles</p>'

**“Lang” Methods**

**\_.clone(value, [isDeep], [customizer], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_clonevalue-isdeep-customizer-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8496) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.clone)

Creates a clone of value. If isDeep is true nested objects are cloned, otherwise they are assigned by reference. Ifcustomizer is provided it's invoked to produce the cloned values. If customizer returns undefined cloning is handled by the method instead. The customizer is bound to thisArg and invoked with up to three argument; (value [, index|key, object]).   
  
**Note:** This method is loosely based on the [structured clone algorithm](http://www.w3.org/TR/html5/infrastructure.html#internal-structured-cloning-algorithm). The enumerable properties of arguments objects and objects created by constructors other than Object are cloned to plain Object objects. An empty object is returned for uncloneable values such as functions, DOM nodes, Maps, Sets, and WeakMaps.

**Arguments**

1. value *(\*)*: The value to clone.
2. [isDeep] *(boolean)*: Specify a deep clone.
3. [customizer] *(Function)*: The function to customize cloning values.
4. [thisArg] *(\*)*: The this binding of customizer.

**Returns**

*(\*)*: Returns the cloned value.

**Example**

var users = [

{ 'user': 'barney' },

{ 'user': 'fred' }

];

var shallow = \_.clone(users);

shallow[0] === users[0];

// => true

var deep = \_.clone(users, true);

deep[0] === users[0];

// => false

// using a customizer callback

var el = \_.clone(document.body, function(value) {

if (\_.isElement(value)) {

return value.cloneNode(false);

}

});

el === document.body

// => false

el.nodeName

// => BODY

el.childNodes.length;

// => 0

**\_.cloneDeep(value, [customizer], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_clonedeepvalue-customizer-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8555) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.clonedeep)

Creates a deep clone of value. If customizer is provided it's invoked to produce the cloned values. If customizer returnsundefined cloning is handled by the method instead. The customizer is bound to thisArg and invoked with up to three argument; (value [, index|key, object]).   
  
**Note:** This method is loosely based on the [structured clone algorithm](http://www.w3.org/TR/html5/infrastructure.html#internal-structured-cloning-algorithm). The enumerable properties of arguments objects and objects created by constructors other than Object are cloned to plain Object objects. An empty object is returned for uncloneable values such as functions, DOM nodes, Maps, Sets, and WeakMaps.

**Arguments**

1. value *(\*)*: The value to deep clone.
2. [customizer] *(Function)*: The function to customize cloning values.
3. [thisArg] *(\*)*: The this binding of customizer.

**Returns**

*(\*)*: Returns the deep cloned value.

**Example**

var users = [

{ 'user': 'barney' },

{ 'user': 'fred' }

];

var deep = \_.cloneDeep(users);

deep[0] === users[0];

// => false

// using a customizer callback

var el = \_.cloneDeep(document.body, function(value) {

if (\_.isElement(value)) {

return value.cloneNode(true);

}

});

el === document.body

// => false

el.nodeName

// => BODY

el.childNodes.length;

// => 20

**\_.gt(value, other)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_gtvalue-other) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8581) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.gt)

Checks if value is greater than other.

**Arguments**

1. value *(\*)*: The value to compare.
2. other *(\*)*: The other value to compare.

**Returns**

*(boolean)*: Returns true if value is greater than other, else false.

**Example**

\_.gt(3, 1);

// => true

\_.gt(3, 3);

// => false

\_.gt(1, 3);

// => false

**\_.gte(value, other)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_gtevalue-other) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8605) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.gte)

Checks if value is greater than or equal to other.

**Arguments**

1. value *(\*)*: The value to compare.
2. other *(\*)*: The other value to compare.

**Returns**

*(boolean)*: Returns true if value is greater than or equal to other, else false.

**Example**

\_.gte(3, 1);

// => true

\_.gte(3, 3);

// => true

\_.gte(1, 3);

// => false

**\_.isArguments(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isargumentsvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8625) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isarguments)

Checks if value is classified as an arguments object.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isArguments(function() { return arguments; }());

// => true

\_.isArguments([1, 2, 3]);

// => false

**\_.isArray(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isarrayvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8646) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isarray)

Checks if value is classified as an Array object.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isArray([1, 2, 3]);

// => true

\_.isArray(function() { return arguments; }());

// => false

**\_.isBoolean(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isbooleanvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8666) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isboolean)

Checks if value is classified as a boolean primitive or object.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isBoolean(false);

// => true

\_.isBoolean(null);

// => false

**\_.isDate(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isdatevalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8686) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isdate)

Checks if value is classified as a Date object.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isDate(new Date);

// => true

\_.isDate('Mon April 23 2012');

// => false

**\_.isElement(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_iselementvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8706) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.iselement)

Checks if value is a DOM element.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is a DOM element, else false.

**Example**

\_.isElement(document.body);

// => true

\_.isElement('<body>');

// => false

**\_.isEmpty(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isemptyvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8737) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isempty)

Checks if value is empty. A value is considered empty unless it's an arguments object, array, string, or jQuery-like collection with a length greater than 0 or an object with own enumerable properties.

**Arguments**

1. value *(Array|Object|string)*: The value to inspect.

**Returns**

*(boolean)*: Returns true if value is empty, else false.

**Example**

\_.isEmpty(null);

// => true

\_.isEmpty(true);

// => true

\_.isEmpty(1);

// => true

\_.isEmpty([1, 2, 3]);

// => false

\_.isEmpty({ 'a': 1 });

// => false

**\_.isEqual(value, other, [customizer], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isequalvalue-other-customizer-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8792) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isequal)

Performs a deep comparison between two values to determine if they are equivalent. If customizer is provided it's invoked to compare values. If customizer returns undefined comparisons are handled by the method instead. The customizer is bound to thisArg and invoked with up to three arguments: (value, other [, index|key]).   
  
**Note:** This method supports comparing arrays, booleans, Date objects, numbers, Object objects, regexes, and strings. Objects are compared by their own, not inherited, enumerable properties. Functions and DOM nodes are **not** supported. Provide a customizer function to extend support for comparing other values.

**Aliases**

*\_.eq*

**Arguments**

1. value *(\*)*: The value to compare.
2. other *(\*)*: The other value to compare.
3. [customizer] *(Function)*: The function to customize value comparisons.
4. [thisArg] *(\*)*: The this binding of customizer.

**Returns**

*(boolean)*: Returns true if the values are equivalent, else false.

**Example**

var object = { 'user': 'fred' };

var other = { 'user': 'fred' };

object == other;

// => false

\_.isEqual(object, other);

// => true

// using a customizer callback

var array = ['hello', 'goodbye'];

var other = ['hi', 'goodbye'];

\_.isEqual(array, other, function(value, other) {

if (\_.every([value, other], RegExp.prototype.test, /^h(?:i|ello)$/)) {

return true;

}

});

// => true

**\_.isError(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_iserrorvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8815) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.iserror)

Checks if value is an Error, EvalError, RangeError, ReferenceError, SyntaxError, TypeError, or URIError object.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is an error object, else false.

**Example**

\_.isError(new Error);

// => true

\_.isError(Error);

// => false

**\_.isFinite(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isfinitevalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8846) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isfinite)

Checks if value is a finite primitive number.   
  
**Note:** This method is based on [Number.isFinite](http://ecma-international.org/ecma-262/6.0/#sec-number.isfinite).

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is a finite number, else false.

**Example**

\_.isFinite(10);

// => true

\_.isFinite('10');

// => false

\_.isFinite(true);

// => false

\_.isFinite(Object(10));

// => false

\_.isFinite(Infinity);

// => false

**\_.isFunction(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isfunctionvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8866) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isfunction)

Checks if value is classified as a Function object.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isFunction(\_);

// => true

\_.isFunction(/abc/);

// => false

**\_.isMatch(object, source, [customizer], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_ismatchobject-source-customizer-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8939) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.ismatch)

Performs a deep comparison between object and source to determine if object contains equivalent property values. Ifcustomizer is provided it's invoked to compare values. If customizer returns undefined comparisons are handled by the method instead. The customizer is bound to thisArg and invoked with three arguments: (value, other, index|key).   
  
**Note:** This method supports comparing properties of arrays, booleans, Date objects, numbers, Object objects, regexes, and strings. Functions and DOM nodes are **not** supported. Provide a customizer function to extend support for comparing other values.

**Arguments**

1. object *(Object)*: The object to inspect.
2. source *(Object)*: The object of property values to match.
3. [customizer] *(Function)*: The function to customize value comparisons.
4. [thisArg] *(\*)*: The this binding of customizer.

**Returns**

*(boolean)*: Returns true if object is a match, else false.

**Example**

var object = { 'user': 'fred', 'age': 40 };

\_.isMatch(object, { 'age': 40 });

// => true

\_.isMatch(object, { 'age': 36 });

// => false

// using a customizer callback

var object = { 'greeting': 'hello' };

var source = { 'greeting': 'hi' };

\_.isMatch(object, source, function(value, other) {

return \_.every([value, other], RegExp.prototype.test, /^h(?:i|ello)$/) || undefined;

});

// => true

**\_.isNaN(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isnanvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8969) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isnan)

Checks if value is NaN.   
  
**Note:** This method is not the same as [isNaN](https://es5.github.io/#x15.1.2.4) which returns true for undefined and other non-numeric values.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is NaN, else false.

**Example**

\_.isNaN(NaN);

// => true

\_.isNaN(new Number(NaN));

// => true

isNaN(undefined);

// => true

\_.isNaN(undefined);

// => false

**\_.isNative(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isnativevalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8991) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isnative)

Checks if value is a native function.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is a native function, else false.

**Example**

\_.isNative(Array.prototype.push);

// => true

\_.isNative(\_);

// => false

**\_.isNull(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isnullvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9017) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isnull)

Checks if value is null.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is null, else false.

**Example**

\_.isNull(null);

// => true

\_.isNull(void 0);

// => false

**\_.isNumber(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isnumbervalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9043) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isnumber)

Checks if value is classified as a Number primitive or object.   
  
**Note:** To exclude Infinity, -Infinity, and NaN, which are classified as numbers, use the \_.isFinite method.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isNumber(8.4);

// => true

\_.isNumber(NaN);

// => true

\_.isNumber('8.4');

// => false

**\_.isObject(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isobjectvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L8893) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isobject)

Checks if value is the [language type](https://es5.github.io/#x8) of Object. (e.g. arrays, functions, objects, regexes, new Number(0), and new String(''))

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is an object, else false.

**Example**

\_.isObject({});

// => true

\_.isObject([1, 2, 3]);

// => true

\_.isObject(1);

// => false

**\_.isPlainObject(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isplainobjectvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9077) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isplainobject)

Checks if value is a plain object, that is, an object created by the Object constructor or one with a [[Prototype]] of null.   
  
**Note:** This method assumes objects created by the Object constructor have no inherited enumerable properties.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is a plain object, else false.

**Example**

function Foo() {

this.a = 1;

}

\_.isPlainObject(new Foo);

// => false

\_.isPlainObject([1, 2, 3]);

// => false

\_.isPlainObject({ 'x': 0, 'y': 0 });

// => true

\_.isPlainObject(Object.create(null));

// => true

**\_.isRegExp(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isregexpvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9121) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isregexp)

Checks if value is classified as a RegExp object.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isRegExp(/abc/);

// => true

\_.isRegExp('/abc/');

// => false

**\_.isString(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isstringvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9141) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isstring)

Checks if value is classified as a String primitive or object.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isString('abc');

// => true

\_.isString(1);

// => false

**\_.isTypedArray(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_istypedarrayvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9161) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.istypedarray)

Checks if value is classified as a typed array.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is correctly classified, else false.

**Example**

\_.isTypedArray(new Uint8Array);

// => true

\_.isTypedArray([]);

// => false

**\_.isUndefined(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_isundefinedvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9181) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.isundefined)

Checks if value is undefined.

**Arguments**

1. value *(\*)*: The value to check.

**Returns**

*(boolean)*: Returns true if value is undefined, else false.

**Example**

\_.isUndefined(void 0);

// => true

\_.isUndefined(null);

// => false

**\_.lt(value, other)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_ltvalue-other) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9205) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.lt)

Checks if value is less than other.

**Arguments**

1. value *(\*)*: The value to compare.
2. other *(\*)*: The other value to compare.

**Returns**

*(boolean)*: Returns true if value is less than other, else false.

**Example**

\_.lt(1, 3);

// => true

\_.lt(3, 3);

// => false

\_.lt(3, 1);

// => false

**\_.lte(value, other)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_ltevalue-other) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9229) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.lte)

Checks if value is less than or equal to other.

**Arguments**

1. value *(\*)*: The value to compare.
2. other *(\*)*: The other value to compare.

**Returns**

*(boolean)*: Returns true if value is less than or equal to other, else false.

**Example**

\_.lte(1, 3);

// => true

\_.lte(3, 3);

// => true

\_.lte(3, 1);

// => false

**\_.toArray(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_toarrayvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9248) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.toarray)

Converts value to an array.

**Arguments**

1. value *(\*)*: The value to convert.

**Returns**

*(Array)*: Returns the converted array.

**Example**

(function() {

return \_.toArray(arguments).slice(1);

}(1, 2, 3));

// => [2, 3]

**\_.toPlainObject(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_toplainobjectvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9284) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.toplainobject)

Converts value to a plain object flattening inherited enumerable properties of value to own properties of the plain object.

**Arguments**

1. value *(\*)*: The value to convert.

**Returns**

*(Object)*: Returns the converted plain object.

**Example**

function Foo() {

this.b = 2;

}

Foo.prototype.c = 3;

\_.assign({ 'a': 1 }, new Foo);

// => { 'a': 1, 'b': 2 }

\_.assign({ 'a': 1 }, \_.toPlainObject(new Foo));

// => { 'a': 1, 'b': 2, 'c': 3 }

**“Math” Methods**

**\_.add(augend, addend)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_addaugend-addend) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11803) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.add)

Adds two numbers.

**Arguments**

1. augend *(number)*: The first number to add.
2. addend *(number)*: The second number to add.

**Returns**

*(number)*: Returns the sum.

**Example**

\_.add(6, 4);

// => 10

**\_.ceil(n, [precision=0])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_ceiln-precision0) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11827) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.ceil)

Calculates n rounded up to precision.

**Arguments**

1. n *(number)*: The number to round up.
2. [precision=0] *(number)*: The precision to round up to.

**Returns**

*(number)*: Returns the rounded up number.

**Example**

\_.ceil(4.006);

// => 5

\_.ceil(6.004, 2);

// => 6.01

\_.ceil(6040, -2);

// => 6100

**\_.floor(n, [precision=0])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_floorn-precision0) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11849) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.floor)

Calculates n rounded down to precision.

**Arguments**

1. n *(number)*: The number to round down.
2. [precision=0] *(number)*: The precision to round down to.

**Returns**

*(number)*: Returns the rounded down number.

**Example**

\_.floor(4.006);

// => 4

\_.floor(0.046, 2);

// => 0.04

\_.floor(4060, -2);

// => 4000

**\_.max(collection, [iteratee], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_maxcollection-iteratee-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11898) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.max)

Gets the maximum value of collection. If collection is empty or falsey -Infinity is returned. If an iteratee function is provided it's invoked for each value in collection to generate the criterion by which the value is ranked. The iteratee is bound to thisArg and invoked with three arguments: (value, index, collection).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(\*)*: Returns the maximum value.

**Example**

\_.max([4, 2, 8, 6]);

// => 8

\_.max([]);

// => -Infinity

var users = [

{ 'user': 'barney', 'age': 36 },

{ 'user': 'fred', 'age': 40 }

];

\_.max(users, function(chr) {

return chr.age;

});

// => { 'user': 'fred', 'age': 40 }

// using the `\_.property` callback shorthand

\_.max(users, 'age');

// => { 'user': 'fred', 'age': 40 }

**\_.min(collection, [iteratee], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_mincollection-iteratee-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11947) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.min)

Gets the minimum value of collection. If collection is empty or falsey Infinity is returned. If an iteratee function is provided it's invoked for each value in collection to generate the criterion by which the value is ranked. The iteratee is bound to thisArg and invoked with three arguments: (value, index, collection).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(\*)*: Returns the minimum value.

**Example**

\_.min([4, 2, 8, 6]);

// => 2

\_.min([]);

// => Infinity

var users = [

{ 'user': 'barney', 'age': 36 },

{ 'user': 'fred', 'age': 40 }

];

\_.min(users, function(chr) {

return chr.age;

});

// => { 'user': 'barney', 'age': 36 }

// using the `\_.property` callback shorthand

\_.min(users, 'age');

// => { 'user': 'barney', 'age': 36 }

**\_.round(n, [precision=0])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_roundn-precision0) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11969) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.round)

Calculates n rounded to precision.

**Arguments**

1. n *(number)*: The number to round.
2. [precision=0] *(number)*: The precision to round to.

**Returns**

*(number)*: Returns the rounded number.

**Example**

\_.round(4.006);

// => 4

\_.round(4.006, 2);

// => 4.01

\_.round(4060, -2);

// => 4100

**\_.sum(collection, [iteratee], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_sumcollection-iteratee-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L12003) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.sum)

Gets the sum of the values in collection.

**Arguments**

1. collection *(Array|Object|string)*: The collection to iterate over.
2. [iteratee] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(number)*: Returns the sum.

**Example**

\_.sum([4, 6]);

// => 10

\_.sum({ 'a': 4, 'b': 6 });

// => 10

var objects = [

{ 'n': 4 },

{ 'n': 6 }

];

\_.sum(objects, function(object) {

return object.n;

});

// => 10

// using the `\_.property` callback shorthand

\_.sum(objects, 'n');

// => 10

**“Number” Methods**

**\_.inRange(n, [start=0], end)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_inrangen-start0-end) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10321) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.inrange)

Checks if n is between start and up to but not including, end. If end is not specified it's set to start with start then set to 0.

**Arguments**

1. n *(number)*: The number to check.
2. [start=0] *(number)*: The start of the range.
3. end *(number)*: The end of the range.

**Returns**

*(boolean)*: Returns true if n is in the range, else false.

**Example**

\_.inRange(3, 2, 4);

// => true

\_.inRange(4, 8);

// => true

\_.inRange(4, 2);

// => false

\_.inRange(2, 2);

// => false

\_.inRange(1.2, 2);

// => true

\_.inRange(5.2, 4);

// => false

**\_.random([min=0], [max=1], [floating])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_randommin0-max1-floating) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10359) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.random)

Produces a random number between min and max (inclusive). If only one argument is provided a number between 0 and the given number is returned. If floating is true, or either min or max are floats, a floating-point number is returned instead of an integer.

**Arguments**

1. [min=0] *(number)*: The minimum possible value.
2. [max=1] *(number)*: The maximum possible value.
3. [floating] *(boolean)*: Specify returning a floating-point number.

**Returns**

*(number)*: Returns the random number.

**Example**

\_.random(0, 5);

// => an integer between 0 and 5

\_.random(5);

// => also an integer between 0 and 5

\_.random(5, true);

// => a floating-point number between 0 and 5

\_.random(1.2, 5.2);

// => a floating-point number between 1.2 and 5.2

**“Object” Methods**

**\_.assign(object, [sources], [customizer], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_assignobject-sources-customizer-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9372) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.assign)

Assigns own enumerable properties of source object(s) to the destination object. Subsequent sources overwrite property assignments of previous sources. If customizer is provided it's invoked to produce the assigned values. The customizer is bound to thisArg and invoked with five arguments:  
(objectValue, sourceValue, key, object, source).   
  
**Note:** This method mutates object and is based on [Object.assign](http://ecma-international.org/ecma-262/6.0/#sec-object.assign).

**Aliases**

*\_.extend*

**Arguments**

1. object *(Object)*: The destination object.
2. [sources] *(...Object)*: The source objects.
3. [customizer] *(Function)*: The function to customize assigned values.
4. [thisArg] *(\*)*: The this binding of customizer.

**Returns**

*(Object)*: Returns object.

**Example**

\_.assign({ 'user': 'barney' }, { 'age': 40 }, { 'user': 'fred' });

// => { 'user': 'fred', 'age': 40 }

// using a customizer callback

var defaults = \_.partialRight(\_.assign, function(value, other) {

return \_.isUndefined(value) ? other : value;

});

defaults({ 'user': 'barney' }, { 'age': 36 }, { 'user': 'fred' });

// => { 'user': 'barney', 'age': 36 }

**\_.create(prototype, [properties])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_createprototype-properties) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9412) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.create)

Creates an object that inherits from the given prototype object. If a properties object is provided its own enumerable properties are assigned to the created object.

**Arguments**

1. prototype *(Object)*: The object to inherit from.
2. [properties] *(Object)*: The properties to assign to the object.

**Returns**

*(Object)*: Returns the new object.

**Example**

function Shape() {

this.x = 0;

this.y = 0;

}

function Circle() {

Shape.call(this);

}

Circle.prototype = \_.create(Shape.prototype, {

'constructor': Circle

});

var circle = new Circle;

circle instanceof Circle;

// => true

circle instanceof Shape;

// => true

**\_.defaults(object, [sources])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_defaultsobject-sources) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9438) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.defaults)

Assigns own enumerable properties of source object(s) to the destination object for all destination properties that resolve toundefined. Once a property is set, additional values of the same property are ignored.   
  
**Note:** This method mutates object.

**Arguments**

1. object *(Object)*: The destination object.
2. [sources] *(...Object)*: The source objects.

**Returns**

*(Object)*: Returns object.

**Example**

\_.defaults({ 'user': 'barney' }, { 'age': 36 }, { 'user': 'fred' });

// => { 'user': 'barney', 'age': 36 }

**\_.defaultsDeep(object, [sources])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_defaultsdeepobject-sources) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9458) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.defaultsdeep)

This method is like \_.defaults except that it recursively assigns default properties.   
  
**Note:** This method mutates object.

**Arguments**

1. object *(Object)*: The destination object.
2. [sources] *(...Object)*: The source objects.

**Returns**

*(Object)*: Returns object.

**Example**

\_.defaultsDeep({ 'user': { 'name': 'barney' } }, { 'user': { 'name': 'fred', 'age': 36 } });

// => { 'user': { 'name': 'barney', 'age': 36 } }

**\_.findKey(object, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_findkeyobject-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9508) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.findkey)

This method is like \_.find except that it returns the key of the first element predicate returns truthy for instead of the element itself.   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. object *(Object)*: The object to search.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(string|undefined)*: Returns the key of the matched element, else undefined.

**Example**

var users = {

'barney': { 'age': 36, 'active': true },

'fred': { 'age': 40, 'active': false },

'pebbles': { 'age': 1, 'active': true }

};

\_.findKey(users, function(chr) {

return chr.age < 40;

});

// => 'barney' (iteration order is not guaranteed)

// using the `\_.matches` callback shorthand

\_.findKey(users, { 'age': 1, 'active': true });

// => 'pebbles'

// using the `\_.matchesProperty` callback shorthand

\_.findKey(users, 'active', false);

// => 'fred'

// using the `\_.property` callback shorthand

\_.findKey(users, 'active');

// => 'barney'

**\_.findLastKey(object, [predicate=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_findlastkeyobject-predicate_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9558) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.findlastkey)

This method is like \_.findKey except that it iterates over elements of a collection in the opposite order.   
  
If a property name is provided for predicate the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for predicate the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. object *(Object)*: The object to search.
2. [predicate=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(string|undefined)*: Returns the key of the matched element, else undefined.

**Example**

var users = {

'barney': { 'age': 36, 'active': true },

'fred': { 'age': 40, 'active': false },

'pebbles': { 'age': 1, 'active': true }

};

\_.findLastKey(users, function(chr) {

return chr.age < 40;

});

// => returns `pebbles` assuming `\_.findKey` returns `barney`

// using the `\_.matches` callback shorthand

\_.findLastKey(users, { 'age': 36, 'active': true });

// => 'barney'

// using the `\_.matchesProperty` callback shorthand

\_.findLastKey(users, 'active', false);

// => 'fred'

// using the `\_.property` callback shorthand

\_.findLastKey(users, 'active');

// => 'pebbles'

**\_.forIn(object, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_forinobject-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9587) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.forin)

Iterates over own and inherited enumerable properties of an object invoking iteratee for each property. The iteratee is bound to thisArg and invoked with three arguments: (value, key, object). Iteratee functions may exit iteration early by explicitly returning false.

**Arguments**

1. object *(Object)*: The object to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns object.

**Example**

function Foo() {

this.a = 1;

this.b = 2;

}

Foo.prototype.c = 3;

\_.forIn(new Foo, function(value, key) {

console.log(key);

});

// => logs 'a', 'b', and 'c' (iteration order is not guaranteed)

**\_.forInRight(object, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_forinrightobject-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9614) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.forinright)

This method is like \_.forIn except that it iterates over properties of object in the opposite order.

**Arguments**

1. object *(Object)*: The object to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns object.

**Example**

function Foo() {

this.a = 1;

this.b = 2;

}

Foo.prototype.c = 3;

\_.forInRight(new Foo, function(value, key) {

console.log(key);

});

// => logs 'c', 'b', and 'a' assuming `\_.forIn ` logs 'a', 'b', and 'c'

**\_.forOwn(object, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_forownobject-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9643) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.forown)

Iterates over own enumerable properties of an object invoking iteratee for each property. The iteratee is bound tothisArg and invoked with three arguments: (value, key, object). Iteratee functions may exit iteration early by explicitly returning false.

**Arguments**

1. object *(Object)*: The object to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns object.

**Example**

function Foo() {

this.a = 1;

this.b = 2;

}

Foo.prototype.c = 3;

\_.forOwn(new Foo, function(value, key) {

console.log(key);

});

// => logs 'a' and 'b' (iteration order is not guaranteed)

**\_.forOwnRight(object, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_forownrightobject-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9670) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.forownright)

This method is like \_.forOwn except that it iterates over properties of object in the opposite order.

**Arguments**

1. object *(Object)*: The object to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns object.

**Example**

function Foo() {

this.a = 1;

this.b = 2;

}

Foo.prototype.c = 3;

\_.forOwnRight(new Foo, function(value, key) {

console.log(key);

});

// => logs 'b' and 'a' assuming `\_.forOwn` logs 'a' and 'b'

**\_.functions(object)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_functionsobject) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9687) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.functions)

Creates an array of function property names from all enumerable properties, own and inherited, of object.

**Aliases**

*\_.methods*

**Arguments**

1. object *(Object)*: The object to inspect.

**Returns**

*(Array)*: Returns the new array of property names.

**Example**

\_.functions(\_);

// => ['after', 'ary', 'assign', ...]

**\_.get(object, path, [defaultValue])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_getobject-path-defaultvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9715) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.get)

Gets the property value at path of object. If the resolved value is undefined the defaultValue is used in its place.

**Arguments**

1. object *(Object)*: The object to query.
2. path *(Array|string)*: The path of the property to get.
3. [defaultValue] *(\*)*: The value returned if the resolved value is undefined.

**Returns**

*(\*)*: Returns the resolved value.

**Example**

var object = { 'a': [{ 'b': { 'c': 3 } }] };

\_.get(object, 'a[0].b.c');

// => 3

\_.get(object, ['a', '0', 'b', 'c']);

// => 3

\_.get(object, 'a.b.c', 'default');

// => 'default'

**\_.has(object, path)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_hasobject-path) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9742) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.has)

Checks if path is a direct property.

**Arguments**

1. object *(Object)*: The object to query.
2. path *(Array|string)*: The path to check.

**Returns**

*(boolean)*: Returns true if path is a direct property, else false.

**Example**

var object = { 'a': { 'b': { 'c': 3 } } };

\_.has(object, 'a');

// => true

\_.has(object, 'a.b.c');

// => true

\_.has(object, ['a', 'b', 'c']);

// => true

**\_.invert(object, [multiValue])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_invertobject-multivalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9783) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.invert)

Creates an object composed of the inverted keys and values of object. If object contains duplicate values, subsequent values overwrite property assignments of previous values unless multiValue is true.

**Arguments**

1. object *(Object)*: The object to invert.
2. [multiValue] *(boolean)*: Allow multiple values per key.

**Returns**

*(Object)*: Returns the new inverted object.

**Example**

var object = { 'a': 1, 'b': 2, 'c': 1 };

\_.invert(object);

// => { '1': 'c', '2': 'b' }

// with `multiValue`

\_.invert(object, true);

// => { '1': ['a', 'c'], '2': ['b'] }

**\_.keys(object)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_keysobject) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9837) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.keys)

Creates an array of the own enumerable property names of object.   
  
**Note:** Non-object values are coerced to objects. See the [ES spec](http://ecma-international.org/ecma-262/6.0/#sec-object.keys) for more details.

**Arguments**

1. object *(Object)*: The object to query.

**Returns**

*(Array)*: Returns the array of property names.

**Example**

function Foo() {

this.a = 1;

this.b = 2;

}

Foo.prototype.c = 3;

\_.keys(new Foo);

// => ['a', 'b'] (iteration order is not guaranteed)

\_.keys('hi');

// => ['0', '1']

**\_.keysIn(object)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_keysinobject) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9868) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.keysin)

Creates an array of the own and inherited enumerable property names of object.   
  
**Note:** Non-object values are coerced to objects.

**Arguments**

1. object *(Object)*: The object to query.

**Returns**

*(Array)*: Returns the array of property names.

**Example**

function Foo() {

this.a = 1;

this.b = 2;

}

Foo.prototype.c = 3;

\_.keysIn(new Foo);

// => ['a', 'b', 'c'] (iteration order is not guaranteed)

**\_.mapKeys(object, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_mapkeysobject-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9945) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.mapkeys)

The opposite of \_.mapValues; this method creates an object with the same values as object and keys generated by running each own enumerable property of object through iteratee.

**Arguments**

1. object *(Object)*: The object to iterate over.
2. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns the new mapped object.

**Example**

\_.mapKeys({ 'a': 1, 'b': 2 }, function(value, key) {

return key + value;

});

// => { 'a1': 1, 'b2': 2 }

**\_.mapValues(object, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_mapvaluesobject-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9988) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.mapvalues)

Creates an object with the same keys as object and values generated by running each own enumerable property of objectthrough iteratee. The iteratee function is bound to thisArg and invoked with three arguments:  
(value, key, object).   
  
If a property name is provided for iteratee the created \_.property style callback returns the property value of the given element.   
  
If a value is also provided for thisArg the created \_.matchesProperty style callback returns true for elements that have a matching property value, else false.   
  
If an object is provided for iteratee the created \_.matches style callback returns true for elements that have the properties of the given object, else false.

**Arguments**

1. object *(Object)*: The object to iterate over.
2. [iteratee=\_.identity] *(Function|Object|string)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Object)*: Returns the new mapped object.

**Example**

\_.mapValues({ 'a': 1, 'b': 2 }, function(n) {

return n \* 3;

});

// => { 'a': 3, 'b': 6 }

var users = {

'fred': { 'user': 'fred', 'age': 40 },

'pebbles': { 'user': 'pebbles', 'age': 1 }

};

// using the `\_.property` callback shorthand

\_.mapValues(users, 'age');

// => { 'fred': 40, 'pebbles': 1 } (iteration order is not guaranteed)

**\_.merge(object, [sources], [customizer], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_mergeobject-sources-customizer-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L9338) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.merge)

Recursively merges own enumerable properties of the source object(s), that don't resolve to undefined into the destination object. Subsequent sources overwrite property assignments of previous sources. If customizer is provided it's invoked to produce the merged values of the destination and source properties. If customizer returns undefined merging is handled by the method instead. The customizer is bound to thisArg and invoked with five arguments: (objectValue, sourceValue, key, object, source).

**Arguments**

1. object *(Object)*: The destination object.
2. [sources] *(...Object)*: The source objects.
3. [customizer] *(Function)*: The function to customize assigned values.
4. [thisArg] *(\*)*: The this binding of customizer.

**Returns**

*(Object)*: Returns object.

**Example**

var users = {

'data': [{ 'user': 'barney' }, { 'user': 'fred' }]

};

var ages = {

'data': [{ 'age': 36 }, { 'age': 40 }]

};

\_.merge(users, ages);

// => { 'data': [{ 'user': 'barney', 'age': 36 }, { 'user': 'fred', 'age': 40 }] }

// using a customizer callback

var object = {

'fruits': ['apple'],

'vegetables': ['beet']

};

var other = {

'fruits': ['banana'],

'vegetables': ['carrot']

};

\_.merge(object, other, function(a, b) {

if (\_.isArray(a)) {

return a.concat(b);

}

});

// => { 'fruits': ['apple', 'banana'], 'vegetables': ['beet', 'carrot'] }

**\_.omit(object, [predicate], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_omitobject-predicate-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10013) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.omit)

The opposite of \_.pick; this method creates an object composed of the own and inherited enumerable properties of objectthat are not omitted.

**Arguments**

1. object *(Object)*: The source object.
2. [predicate] *(Function|...(string|string[])*: The function invoked per iteration or property names to omit, specified as individual property names or arrays of property names.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Object)*: Returns the new object.

**Example**

var object = { 'user': 'fred', 'age': 40 };

\_.omit(object, 'age');

// => { 'user': 'fred' }

\_.omit(object, \_.isNumber);

// => { 'user': 'fred' }

**\_.pairs(object)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_pairsobject) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10041) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.pairs)

Creates a two dimensional array of the key-value pairs for object, e.g. [[key1, value1], [key2, value2]].

**Arguments**

1. object *(Object)*: The object to query.

**Returns**

*(Array)*: Returns the new array of key-value pairs.

**Example**

\_.pairs({ 'barney': 36, 'fred': 40 });

// => [['barney', 36], ['fred', 40]] (iteration order is not guaranteed)

**\_.pick(object, [predicate], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_pickobject-predicate-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10082) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.pick)

Creates an object composed of the picked object properties. Property names may be specified as individual arguments or as arrays of property names. If predicate is provided it's invoked for each property of object picking the propertiespredicate returns truthy for. The predicate is bound to thisArg and invoked with three arguments: (value, key, object).

**Arguments**

1. object *(Object)*: The source object.
2. [predicate] *(Function|...(string|string[])*: The function invoked per iteration or property names to pick, specified as individual property names or arrays of property names.
3. [thisArg] *(\*)*: The this binding of predicate.

**Returns**

*(Object)*: Returns the new object.

**Example**

var object = { 'user': 'fred', 'age': 40 };

\_.pick(object, 'user');

// => { 'user': 'fred' }

\_.pick(object, \_.isString);

// => { 'user': 'fred' }

**\_.result(object, path, [defaultValue])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_resultobject-path-defaultvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10119) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.result)

This method is like \_.get except that if the resolved value is a function it's invoked with the this binding of its parent object and its result is returned.

**Arguments**

1. object *(Object)*: The object to query.
2. path *(Array|string)*: The path of the property to resolve.
3. [defaultValue] *(\*)*: The value returned if the resolved value is undefined.

**Returns**

*(\*)*: Returns the resolved value.

**Example**

var object = { 'a': [{ 'b': { 'c1': 3, 'c2': \_.constant(4) } }] };

\_.result(object, 'a[0].b.c1');

// => 3

\_.result(object, 'a[0].b.c2');

// => 4

\_.result(object, 'a.b.c', 'default');

// => 'default'

\_.result(object, 'a.b.c', \_.constant('default'));

// => 'default'

**\_.set(object, path, value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_setobject-path-value) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10155) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.set)

Sets the property value of path on object. If a portion of path does not exist it's created.

**Arguments**

1. object *(Object)*: The object to augment.
2. path *(Array|string)*: The path of the property to set.
3. value *(\*)*: The value to set.

**Returns**

*(Object)*: Returns object.

**Example**

var object = { 'a': [{ 'b': { 'c': 3 } }] };

\_.set(object, 'a[0].b.c', 4);

console.log(object.a[0].b.c);

// => 4

\_.set(object, 'x[0].y.z', 5);

console.log(object.x[0].y.z);

// => 5

**\_.transform(object, [iteratee=\_.identity], [accumulator], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_transformobject-iteratee_identity-accumulator-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10210) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.transform)

An alternative to \_.reduce; this method transforms object to a new accumulator object which is the result of running each of its own enumerable properties through iteratee, with each invocation potentially mutating the accumulator object. Theiteratee is bound to thisArg and invoked with four arguments: (accumulator, value, key, object). Iteratee functions may exit iteration early by explicitly returning false.

**Arguments**

1. object *(Array|Object)*: The object to iterate over.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [accumulator] *(\*)*: The custom accumulator value.
4. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(\*)*: Returns the accumulated value.

**Example**

\_.transform([2, 3, 4], function(result, n) {

result.push(n \*= n);

return n % 2 == 0;

});

// => [4, 9]

\_.transform({ 'a': 1, 'b': 2 }, function(result, n, key) {

result[key] = n \* 3;

});

// => { 'a': 3, 'b': 6 }

**\_.values(object)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_valuesobject) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10257) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.values)

Creates an array of the own enumerable property values of object.   
  
**Note:** Non-object values are coerced to objects.

**Arguments**

1. object *(Object)*: The object to query.

**Returns**

*(Array)*: Returns the array of property values.

**Example**

function Foo() {

this.a = 1;

this.b = 2;

}

Foo.prototype.c = 3;

\_.values(new Foo);

// => [1, 2] (iteration order is not guaranteed)

\_.values('hi');

// => ['h', 'i']

**\_.valuesIn(object)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_valuesinobject) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10284) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.valuesin)

Creates an array of the own and inherited enumerable property values of object.   
  
**Note:** Non-object values are coerced to objects.

**Arguments**

1. object *(Object)*: The object to query.

**Returns**

*(Array)*: Returns the array of property values.

**Example**

function Foo() {

this.a = 1;

this.b = 2;

}

Foo.prototype.c = 3;

\_.valuesIn(new Foo);

// => [1, 2, 3] (iteration order is not guaranteed)

**“String” Methods**

**\_.camelCase([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_camelcasestring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10415) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.camelcase)

Converts string to [camel case](https://en.wikipedia.org/wiki/CamelCase).

**Arguments**

1. [string=''] *(string)*: The string to convert.

**Returns**

*(string)*: Returns the camel cased string.

**Example**

\_.camelCase('Foo Bar');

// => 'fooBar'

\_.camelCase('--foo-bar');

// => 'fooBar'

\_.camelCase('\_\_foo\_bar\_\_');

// => 'fooBar'

**\_.capitalize([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_capitalizestring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10433) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.capitalize)

Capitalizes the first character of string.

**Arguments**

1. [string=''] *(string)*: The string to capitalize.

**Returns**

*(string)*: Returns the capitalized string.

**Example**

\_.capitalize('fred');

// => 'Fred'

**\_.deburr([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_deburrstring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10452) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.deburr)

Deburrs string by converting [latin-1 supplementary letters](https://en.wikipedia.org/wiki/Latin-1_Supplement_(Unicode_block)#Character_table) to basic latin letters and removing [combining diacritical marks](https://en.wikipedia.org/wiki/Combining_Diacritical_Marks).

**Arguments**

1. [string=''] *(string)*: The string to deburr.

**Returns**

*(string)*: Returns the deburred string.

**Example**

\_.deburr('déjà vu');

// => 'deja vu'

**\_.endsWith([string=''], [target], [position=string.length])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_endswithstring-target-positionstringlength) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10478) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.endswith)

Checks if string ends with the given target string.

**Arguments**

1. [string=''] *(string)*: The string to search.
2. [target] *(string)*: The string to search for.
3. [position=string.length] *(number)*: The position to search from.

**Returns**

*(boolean)*: Returns true if string ends with target, else false.

**Example**

\_.endsWith('abc', 'c');

// => true

\_.endsWith('abc', 'b');

// => false

\_.endsWith('abc', 'b', 2);

// => true

**\_.escape([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_escapestring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10523) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.escape)

Converts the characters "&", "<", ">", '"', "'", and "`", in string to their corresponding HTML entities.   
  
**Note:** No other characters are escaped. To escape additional characters use a third-party library like [*he*](https://mths.be/he).   
  
Though the ">" character is escaped for symmetry, characters like ">" and "/" don't need escaping in HTML and have no special meaning unless they're part of a tag or unquoted attribute value. See [Mathias Bynens's article](https://mathiasbynens.be/notes/ambiguous-ampersands) (under "semi-related fun fact") for more details.   
  
Backticks are escaped because in Internet Explorer < 9, they can break out of attribute values or HTML comments. See [#59](https://html5sec.org/#59),[#102](https://html5sec.org/#102), [#108](https://html5sec.org/#108), and [#133](https://html5sec.org/#133) of the [HTML5 Security Cheatsheet](https://html5sec.org/) for more details.   
  
When working with HTML you should always [quote attribute values](http://wonko.com/post/html-escaping) to reduce XSS vectors.

**Arguments**

1. [string=''] *(string)*: The string to escape.

**Returns**

*(string)*: Returns the escaped string.

**Example**

\_.escape('fred, barney, & pebbles');

// => 'fred, barney, &amp; pebbles'

**\_.escapeRegExp([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_escaperegexpstring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10545) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.escaperegexp)

Escapes the RegExp special characters "\", "/", "^", "$", ".", "|", "?", "\*", "+", "(", ")", "[", "]", "{" and "}" in string.

**Arguments**

1. [string=''] *(string)*: The string to escape.

**Returns**

*(string)*: Returns the escaped string.

**Example**

\_.escapeRegExp('[lodash](https://lodash.com/)');

// => '\[lodash\]\(https:\/\/lodash\.com\/\)'

**\_.kebabCase([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_kebabcasestring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10571) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.kebabcase)

Converts string to [kebab case](https://en.wikipedia.org/wiki/Letter_case#Special_case_styles).

**Arguments**

1. [string=''] *(string)*: The string to convert.

**Returns**

*(string)*: Returns the kebab cased string.

**Example**

\_.kebabCase('Foo Bar');

// => 'foo-bar'

\_.kebabCase('fooBar');

// => 'foo-bar'

\_.kebabCase('\_\_foo\_bar\_\_');

// => 'foo-bar'

**\_.pad([string=''], [length=0], [chars=' '])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_padstring-length0-chars) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10597) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.pad)

Pads string on the left and right sides if it's shorter than length. Padding characters are truncated if they can't be evenly divided by length.

**Arguments**

1. [string=''] *(string)*: The string to pad.
2. [length=0] *(number)*: The padding length.
3. [chars=' '] *(string)*: The string used as padding.

**Returns**

*(string)*: Returns the padded string.

**Example**

\_.pad('abc', 8);

// => ' abc '

\_.pad('abc', 8, '\_-');

// => '\_-abc\_-\_'

\_.pad('abc', 3);

// => 'abc'

**\_.padLeft([string=''], [length=0], [chars=' '])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_padleftstring-length0-chars) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10635) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.padleft)

Pads string on the left side if it's shorter than length. Padding characters are truncated if they exceed length.

**Arguments**

1. [string=''] *(string)*: The string to pad.
2. [length=0] *(number)*: The padding length.
3. [chars=' '] *(string)*: The string used as padding.

**Returns**

*(string)*: Returns the padded string.

**Example**

\_.padLeft('abc', 6);

// => ' abc'

\_.padLeft('abc', 6, '\_-');

// => '\_-\_abc'

\_.padLeft('abc', 3);

// => 'abc'

**\_.padRight([string=''], [length=0], [chars=' '])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_padrightstring-length0-chars) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10659) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.padright)

Pads string on the right side if it's shorter than length. Padding characters are truncated if they exceed length.

**Arguments**

1. [string=''] *(string)*: The string to pad.
2. [length=0] *(number)*: The padding length.
3. [chars=' '] *(string)*: The string used as padding.

**Returns**

*(string)*: Returns the padded string.

**Example**

\_.padRight('abc', 6);

// => 'abc '

\_.padRight('abc', 6, '\_-');

// => 'abc\_-\_'

\_.padRight('abc', 3);

// => 'abc'

**\_.parseInt(string, [radix])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_parseintstring-radix) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10684) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.parseint)

Converts string to an integer of the specified radix. If radix is undefined or 0, a radix of 10 is used unless value is a hexadecimal, in which case a radix of 16 is used.   
  
**Note:** This method aligns with the [ES5 implementation](https://es5.github.io/#E) of parseInt.

**Arguments**

1. string *(string)*: The string to convert.
2. [radix] *(number)*: The radix to interpret value by.

**Returns**

*(number)*: Returns the converted integer.

**Example**

\_.parseInt('08');

// => 8

\_.map(['6', '08', '10'], \_.parseInt);

// => [6, 8, 10]

**\_.repeat([string=''], [n=0])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_repeatstring-n0) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10717) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.repeat)

Repeats the given string n times.

**Arguments**

1. [string=''] *(string)*: The string to repeat.
2. [n=0] *(number)*: The number of times to repeat the string.

**Returns**

*(string)*: Returns the repeated string.

**Example**

\_.repeat('\*', 3);

// => '\*\*\*'

\_.repeat('abc', 2);

// => 'abcabc'

\_.repeat('abc', 0);

// => ''

**\_.snakeCase([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_snakecasestring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10756) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.snakecase)

Converts string to [snake case](https://en.wikipedia.org/wiki/Snake_case).

**Arguments**

1. [string=''] *(string)*: The string to convert.

**Returns**

*(string)*: Returns the snake cased string.

**Example**

\_.snakeCase('Foo Bar');

// => 'foo\_bar'

\_.snakeCase('fooBar');

// => 'foo\_bar'

\_.snakeCase('--foo-bar');

// => 'foo\_bar'

**\_.startCase([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_startcasestring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10779) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.startcase)

Converts string to [start case](https://en.wikipedia.org/wiki/Letter_case#Stylistic_or_specialised_usage).

**Arguments**

1. [string=''] *(string)*: The string to convert.

**Returns**

*(string)*: Returns the start cased string.

**Example**

\_.startCase('--foo-bar');

// => 'Foo Bar'

\_.startCase('fooBar');

// => 'Foo Bar'

\_.startCase('\_\_foo\_bar\_\_');

// => 'Foo Bar'

**\_.startsWith([string=''], [target], [position=0])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_startswithstring-target-position0) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10804) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.startswith)

Checks if string starts with the given target string.

**Arguments**

1. [string=''] *(string)*: The string to search.
2. [target] *(string)*: The string to search for.
3. [position=0] *(number)*: The position to search from.

**Returns**

*(boolean)*: Returns true if string starts with target, else false.

**Example**

\_.startsWith('abc', 'a');

// => true

\_.startsWith('abc', 'b');

// => false

\_.startsWith('abc', 'b', 1);

// => true

**\_.template([string=''], [options])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatestring-options) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L10909) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.template)

Creates a compiled template function that can interpolate data properties in "interpolate" delimiters, HTML-escape interpolated data properties in "escape" delimiters, and execute JavaScript in "evaluate" delimiters. Data properties may be accessed as free variables in the template. If a setting object is provided it takes precedence over \_.templateSettingsvalues.   
  
**Note:** In the development build \_.template utilizes [sourceURLs](http://www.html5rocks.com/en/tutorials/developertools/sourcemaps/#toc-sourceurl) for easier debugging.   
  
For more information on precompiling templates see [lodash's custom builds documentation](https://lodash.com/custom-builds).   
  
For more information on Chrome extension sandboxes see [Chrome's extensions documentation](https://developer.chrome.com/extensions/sandboxingEval).

**Arguments**

1. [string=''] *(string)*: The template string.
2. [options] *(Object)*: The options object.
3. [options.escape] *(RegExp)*: The HTML "escape" delimiter.
4. [options.evaluate] *(RegExp)*: The "evaluate" delimiter.
5. [options.imports] *(Object)*: An object to import into the template as free variables.
6. [options.interpolate] *(RegExp)*: The "interpolate" delimiter.
7. [options.sourceURL] *(string)*: The sourceURL of the template's compiled source.
8. [options.variable] *(string)*: The data object variable name.

**Returns**

*(Function)*: Returns the compiled template function.

**Example**

// using the "interpolate" delimiter to create a compiled template

var compiled = \_.template('hello <%= user %>!');

compiled({ 'user': 'fred' });

// => 'hello fred!'

// using the HTML "escape" delimiter to escape data property values

var compiled = \_.template('<b><%- value %></b>');

compiled({ 'value': '<script>' });

// => '<b>&lt;script&gt;</b>'

// using the "evaluate" delimiter to execute JavaScript and generate HTML

var compiled = \_.template('<% \_.forEach(users, function(user) { %><li><%- user %></li><% }); %>');

compiled({ 'users': ['fred', 'barney'] });

// => '<li>fred</li><li>barney</li>'

// using the internal `print` function in "evaluate" delimiters

var compiled = \_.template('<% print("hello " + user); %>!');

compiled({ 'user': 'barney' });

// => 'hello barney!'

// using the ES delimiter as an alternative to the default "interpolate" delimiter

var compiled = \_.template('hello ${ user }!');

compiled({ 'user': 'pebbles' });

// => 'hello pebbles!'

// using custom template delimiters

\_.templateSettings.interpolate = /{{([\s\S]+?)}}/g;

var compiled = \_.template('hello {{ user }}!');

compiled({ 'user': 'mustache' });

// => 'hello mustache!'

// using backslashes to treat delimiters as plain text

var compiled = \_.template('<%= "\\<%- value %\\>" %>');

compiled({ 'value': 'ignored' });

// => '<%- value %>'

// using the `imports` option to import `jQuery` as `jq`

var text = '<% jq.each(users, function(user) { %><li><%- user %></li><% }); %>';

var compiled = \_.template(text, { 'imports': { 'jq': jQuery } });

compiled({ 'users': ['fred', 'barney'] });

// => '<li>fred</li><li>barney</li>'

// using the `sourceURL` option to specify a custom sourceURL for the template

var compiled = \_.template('hello <%= user %>!', { 'sourceURL': '/basic/greeting.jst' });

compiled(data);

// => find the source of "greeting.jst" under the Sources tab or Resources panel of the web inspector

// using the `variable` option to ensure a with-statement isn't used in the compiled template

var compiled = \_.template('hi <%= data.user %>!', { 'variable': 'data' });

compiled.source;

// => function(data) {

// var \_\_t, \_\_p = '';

// \_\_p += 'hi ' + ((\_\_t = ( data.user )) == null ? '' : \_\_t) + '!';

// return \_\_p;

// }

// using the `source` property to inline compiled templates for meaningful

// line numbers in error messages and a stack trace

fs.writeFileSync(path.join(cwd, 'jst.js'), '\

var JST = {\

"main": ' + \_.template(mainText).source + '\

};\

');

**\_.trim([string=''], [chars=whitespace])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_trimstring-charswhitespace) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11036) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.trim)

Removes leading and trailing whitespace or specified characters from string.

**Arguments**

1. [string=''] *(string)*: The string to trim.
2. [chars=whitespace] *(string)*: The characters to trim.

**Returns**

*(string)*: Returns the trimmed string.

**Example**

\_.trim(' abc ');

// => 'abc'

\_.trim('-\_-abc-\_-', '\_-');

// => 'abc'

\_.map([' foo ', ' bar '], \_.trim);

// => ['foo', 'bar']

**\_.trimLeft([string=''], [chars=whitespace])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_trimleftstring-charswhitespace) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11067) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.trimleft)

Removes leading whitespace or specified characters from string.

**Arguments**

1. [string=''] *(string)*: The string to trim.
2. [chars=whitespace] *(string)*: The characters to trim.

**Returns**

*(string)*: Returns the trimmed string.

**Example**

\_.trimLeft(' abc ');

// => 'abc '

\_.trimLeft('-\_-abc-\_-', '\_-');

// => 'abc-\_-'

**\_.trimRight([string=''], [chars=whitespace])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_trimrightstring-charswhitespace) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11097) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.trimright)

Removes trailing whitespace or specified characters from string.

**Arguments**

1. [string=''] *(string)*: The string to trim.
2. [chars=whitespace] *(string)*: The characters to trim.

**Returns**

*(string)*: Returns the trimmed string.

**Example**

\_.trimRight(' abc ');

// => ' abc'

\_.trimRight('-\_-abc-\_-', '\_-');

// => '-\_-abc'

**\_.trunc([string=''], [options], [options.length=30], [options.omission='...'], [options.separator])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_truncstring-options-optionslength30-optionsomission-optionsseparator) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11149) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.trunc)

Truncates string if it's longer than the given maximum string length. The last characters of the truncated string are replaced with the omission string which defaults to "...".

**Arguments**

1. [string=''] *(string)*: The string to truncate.
2. [options] *(Object|number)*: The options object or maximum string length.
3. [options.length=30] *(number)*: The maximum string length.
4. [options.omission='...'] *(string)*: The string to indicate text is omitted.
5. [options.separator] *(RegExp|string)*: The separator pattern to truncate to.

**Returns**

*(string)*: Returns the truncated string.

**Example**

\_.trunc('hi-diddly-ho there, neighborino');

// => 'hi-diddly-ho there, neighbo...'

\_.trunc('hi-diddly-ho there, neighborino', 24);

// => 'hi-diddly-ho there, n...'

\_.trunc('hi-diddly-ho there, neighborino', {

'length': 24,

'separator': ' '

});

// => 'hi-diddly-ho there,...'

\_.trunc('hi-diddly-ho there, neighborino', {

'length': 24,

'separator': /,? +/

});

// => 'hi-diddly-ho there...'

\_.trunc('hi-diddly-ho there, neighborino', {

'omission': ' [...]'

});

// => 'hi-diddly-ho there, neig [...]'

**\_.unescape([string=''])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_unescapestring) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11219) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.unescape)

The inverse of \_.escape; this method converts the HTML entities &amp;, &lt;, &gt;, &quot;, &#39;, and &#96; instring to their corresponding characters.   
  
**Note:** No other HTML entities are unescaped. To unescape additional HTML entities use a third-party library like [*he*](https://mths.be/he).

**Arguments**

1. [string=''] *(string)*: The string to unescape.

**Returns**

*(string)*: Returns the unescaped string.

**Example**

\_.unescape('fred, barney, &amp; pebbles');

// => 'fred, barney, & pebbles'

**\_.words([string=''], [pattern])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_wordsstring-pattern) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11244) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.words)

Splits string into an array of its words.

**Arguments**

1. [string=''] *(string)*: The string to inspect.
2. [pattern] *(RegExp|string)*: The pattern to match words.

**Returns**

*(Array)*: Returns the words of string.

**Example**

\_.words('fred, barney, & pebbles');

// => ['fred', 'barney', 'pebbles']

\_.words('fred, barney, & pebbles', /[^, ]+/g);

// => ['fred', 'barney', '&', 'pebbles']

**“Utility” Methods**

**\_.attempt(func)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_attemptfunc) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11274) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.attempt)

Attempts to invoke func, returning either the result or the caught error object. Any additional arguments are provided tofunc when it's invoked.

**Arguments**

1. func *(Function)*: The function to attempt.

**Returns**

*(\*)*: Returns the func result or error object.

**Example**

// avoid throwing errors for invalid selectors

var elements = \_.attempt(function(selector) {

return document.querySelectorAll(selector);

}, '>\_>');

if (\_.isError(elements)) {

elements = [];

}

**\_.callback([func=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_callbackfunc_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11320) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.callback)

Creates a function that invokes func with the this binding of thisArg and arguments of the created function. If func is a property name the created callback returns the property value for a given element. If func is an object the created callback returns true for elements that contain the equivalent object properties, otherwise it returns false.

**Aliases**

*\_.iteratee*

**Arguments**

1. [func=\_.identity] *(\*)*: The value to convert to a callback.
2. [thisArg] *(\*)*: The this binding of func.

**Returns**

*(Function)*: Returns the callback.

**Example**

var users = [

{ 'user': 'barney', 'age': 36 },

{ 'user': 'fred', 'age': 40 }

];

// wrap to create custom callback shorthands

\_.callback = \_.wrap(\_.callback, function(callback, func, thisArg) {

var match = /^(.+?)\_\_([gl]t)(.+)$/.exec(func);

if (!match) {

return callback(func, thisArg);

}

return function(object) {

return match[2] == 'gt'

? object[match[1]] > match[3]

: object[match[1]] < match[3];

};

});

\_.filter(users, 'age\_\_gt36');

// => [{ 'user': 'fred', 'age': 40 }]

**\_.constant(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_constantvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11345) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.constant)

Creates a function that returns value.

**Arguments**

1. value *(\*)*: The value to return from the new function.

**Returns**

*(Function)*: Returns the new function.

**Example**

var object = { 'user': 'fred' };

var getter = \_.constant(object);

getter() === object;

// => true

**\_.identity(value)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_identityvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11366) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.identity)

This method returns the first argument provided to it.

**Arguments**

1. value *(\*)*: Any value.

**Returns**

*(\*)*: Returns value.

**Example**

var object = { 'user': 'fred' };

\_.identity(object) === object;

// => true

**\_.matches(source)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_matchessource) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11395) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.matches)

Creates a function that performs a deep comparison between a given object and source, returning true if the given object has equivalent property values, else false.   
  
**Note:** This method supports comparing arrays, booleans, Date objects, numbers, Object objects, regexes, and strings. Objects are compared by their own, not inherited, enumerable properties. For comparing a single own or inherited property value see \_.matchesProperty.

**Arguments**

1. source *(Object)*: The object of property values to match.

**Returns**

*(Function)*: Returns the new function.

**Example**

var users = [

{ 'user': 'barney', 'age': 36, 'active': true },

{ 'user': 'fred', 'age': 40, 'active': false }

];

\_.filter(users, \_.matches({ 'age': 40, 'active': false }));

// => [{ 'user': 'fred', 'age': 40, 'active': false }]

**\_.matchesProperty(path, srcValue)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_matchespropertypath-srcvalue) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11423) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.matchesproperty)

Creates a function that compares the property value of path on a given object to value.   
  
**Note:** This method supports comparing arrays, booleans, Date objects, numbers, Object objects, regexes, and strings. Objects are compared by their own, not inherited, enumerable properties.

**Arguments**

1. path *(Array|string)*: The path of the property to get.
2. srcValue *(\*)*: The value to match.

**Returns**

*(Function)*: Returns the new function.

**Example**

var users = [

{ 'user': 'barney' },

{ 'user': 'fred' }

];

\_.find(users, \_.matchesProperty('user', 'fred'));

// => { 'user': 'fred' }

**\_.method(path, [args])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_methodpath-args) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11450) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.method)

Creates a function that invokes the method at path on a given object. Any additional arguments are provided to the invoked method.

**Arguments**

1. path *(Array|string)*: The path of the method to invoke.
2. [args] *(...\*)*: The arguments to invoke the method with.

**Returns**

*(Function)*: Returns the new function.

**Example**

var objects = [

{ 'a': { 'b': { 'c': \_.constant(2) } } },

{ 'a': { 'b': { 'c': \_.constant(1) } } }

];

\_.map(objects, \_.method('a.b.c'));

// => [2, 1]

\_.invoke(\_.sortBy(objects, \_.method(['a', 'b', 'c'])), 'a.b.c');

// => [1, 2]

**\_.methodOf(object, [args])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_methodofobject-args) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11478) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.methodof)

The opposite of \_.method; this method creates a function that invokes the method at a given path on object. Any additional arguments are provided to the invoked method.

**Arguments**

1. object *(Object)*: The object to query.
2. [args] *(...\*)*: The arguments to invoke the method with.

**Returns**

*(Function)*: Returns the new function.

**Example**

var array = \_.times(3, \_.constant),

object = { 'a': array, 'b': array, 'c': array };

\_.map(['a[2]', 'c[0]'], \_.methodOf(object));

// => [2, 0]

\_.map([['a', '2'], ['c', '0']], \_.methodOf(object));

// => [2, 0]

**\_.mixin([object=lodash], source, [options])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_mixinobjectlodash-source-options) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11520) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.mixin)

Adds all own enumerable function properties of a source object to the destination object. If object is a function then methods are added to its prototype as well.   
  
**Note:** Use \_.runInContext to create a pristine lodash function to avoid conflicts caused by modifying the original.

**Arguments**

1. [object=lodash] *(Function|Object)*: The destination object.
2. source *(Object)*: The object of functions to add.
3. [options] *(Object)*: The options object.
4. [options.chain=true] *(boolean)*: Specify whether the functions added are chainable.

**Returns**

*(Function|Object)*: Returns object.

**Example**

function vowels(string) {

return \_.filter(string, function(v) {

return /[aeiou]/i.test(v);

});

}

\_.mixin({ 'vowels': vowels });

\_.vowels('fred');

// => ['e']

\_('fred').vowels().value();

// => ['e']

\_.mixin({ 'vowels': vowels }, { 'chain': false });

\_('fred').vowels();

// => ['e']

**\_.noConflict()**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_noconflict) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11583) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.noconflict)

Reverts the \_ variable to its previous value and returns a reference to the lodash function.

**Returns**

*(Function)*: Returns the lodash function.

**Example**

var lodash = \_.noConflict();

**\_.noop()**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_noop) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11602) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.noop)

A no-operation function that returns undefined regardless of the arguments it receives.

**Example**

var object = { 'user': 'fred' };

\_.noop(object) === undefined;

// => true

**\_.property(path)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_propertypath) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11628) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.property)

Creates a function that returns the property value at path on a given object.

**Arguments**

1. path *(Array|string)*: The path of the property to get.

**Returns**

*(Function)*: Returns the new function.

**Example**

var objects = [

{ 'a': { 'b': { 'c': 2 } } },

{ 'a': { 'b': { 'c': 1 } } }

];

\_.map(objects, \_.property('a.b.c'));

// => [2, 1]

\_.pluck(\_.sortBy(objects, \_.property(['a', 'b', 'c'])), 'a.b.c');

// => [1, 2]

**\_.propertyOf(object)**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_propertyofobject) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11652) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.propertyof)

The opposite of \_.property; this method creates a function that returns the property value at a given path on object.

**Arguments**

1. object *(Object)*: The object to query.

**Returns**

*(Function)*: Returns the new function.

**Example**

var array = [0, 1, 2],

object = { 'a': array, 'b': array, 'c': array };

\_.map(['a[2]', 'c[0]'], \_.propertyOf(object));

// => [2, 0]

\_.map([['a', '2'], ['c', '0']], \_.propertyOf(object));

// => [2, 0]

**\_.range([start=0], end, [step=1])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_rangestart0-end-step1) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11691) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.range)

Creates an array of numbers (positive and/or negative) progressing from start up to, but not including, end. If end is not specified it's set to start with start then set to 0. If end is less than start a zero-length range is created unless a negative step is specified.

**Arguments**

1. [start=0] *(number)*: The start of the range.
2. end *(number)*: The end of the range.
3. [step=1] *(number)*: The value to increment or decrement by.

**Returns**

*(Array)*: Returns the new array of numbers.

**Example**

\_.range(4);

// => [0, 1, 2, 3]

\_.range(1, 5);

// => [1, 2, 3, 4]

\_.range(0, 20, 5);

// => [0, 5, 10, 15]

\_.range(0, -4, -1);

// => [0, -1, -2, -3]

\_.range(1, 4, 0);

// => [1, 1, 1]

\_.range(0);

// => []

**\_.runInContext([context=root])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_runincontextcontextroot) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L723) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.runincontext)

Create a new pristine lodash function using the given context object.

**Arguments**

1. [context=root] *(Object)*: The context object.

**Returns**

*(Function)*: Returns a new lodash function.

**Example**

\_.mixin({ 'foo': \_.constant('foo') });

var lodash = \_.runInContext();

lodash.mixin({ 'bar': lodash.constant('bar') });

\_.isFunction(\_.foo);

// => true

\_.isFunction(\_.bar);

// => false

lodash.isFunction(lodash.foo);

// => false

lodash.isFunction(lodash.bar);

// => true

// using `context` to mock `Date#getTime` use in `\_.now`

var mock = \_.runInContext({

'Date': function() {

return { 'getTime': getTimeMock };

}

});

// or creating a suped-up `defer` in Node.js

var defer = \_.runInContext({ 'setTimeout': setImmediate }).defer;

**\_.times(n, [iteratee=\_.identity], [thisArg])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_timesn-iteratee_identity-thisarg) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11744) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.times)

Invokes the iteratee function n times, returning an array of the results of each invocation. The iteratee is bound tothisArg and invoked with one argument; (index).

**Arguments**

1. n *(number)*: The number of times to invoke iteratee.
2. [iteratee=\_.identity] *(Function)*: The function invoked per iteration.
3. [thisArg] *(\*)*: The this binding of iteratee.

**Returns**

*(Array)*: Returns the array of results.

**Example**

var diceRolls = \_.times(3, \_.partial(\_.random, 1, 6, false));

// => [3, 6, 4]

\_.times(3, function(n) {

mage.castSpell(n);

});

// => invokes `mage.castSpell(n)` three times with `n` of `0`, `1`, and `2`

\_.times(3, function(n) {

this.cast(n);

}, mage);

// => also invokes `mage.castSpell(n)` three times

**\_.uniqueId([prefix])**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_uniqueidprefix) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L11782) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.uniqueid)

Generates a unique ID. If prefix is provided the ID is appended to it.

**Arguments**

1. [prefix] *(string)*: The value to prefix the ID with.

**Returns**

*(string)*: Returns the unique ID.

**Example**

\_.uniqueId('contact\_');

// => 'contact\_104'

\_.uniqueId();

// => '105'

**Methods**

**\_.templateSettings.imports.\_**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsimports_) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1122) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

A reference to the lodash function.

**Properties**

**\_.VERSION**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_version) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L12300) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(string): The semantic version number.

**\_.support**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_support) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L986) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.support)

(Object): An object environment feature flags.

**\_.support.enumErrorProps**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportenumerrorprops) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1003) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(boolean): Detect if name or message properties of Error.prototype are enumerable by default (IE < 9, Safari < 5.1).

**\_.support.enumPrototypes**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportenumprototypes) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1017) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(boolean): Detect if prototype properties are enumerable by default.   
  
Firefox < 3.6, Opera > 9.50 - Opera < 11.60, and Safari < 5.1 (if the prototype or a property on the prototype has been set) incorrectly set the [[Enumerable]] value of a function's prototype property to true.

**\_.support.nonEnumShadows**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportnonenumshadows) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1028) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(boolean): Detect if properties shadowing those on Object.prototype are non-enumerable.   
  
In IE < 9 an object's own properties, shadowing non-enumerable ones, are made non-enumerable as well (a.k.a the JScript[[DontEnum]] bug).

**\_.support.ownLast**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportownlast) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1036) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(boolean): Detect if own properties are iterated after inherited properties (IE < 9).

**\_.support.spliceObjects**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportspliceobjects) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1051) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(boolean): Detect if Array#shift and Array#splice augment array-like objects correctly.   
  
Firefox < 10, compatibility modes of IE 8, and IE < 9 have buggy Array shift() and splice() functions that fail to remove the last element, value[0], of array-like objects even though the "length" property is set to 0. The shift() method is buggy in compatibility modes of IE 8, while splice() is buggy regardless of mode in IE < 9.

**\_.support.unindexedChars**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_supportunindexedchars) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1062) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(boolean): Detect lack of support for accessing string characters by index.   
  
IE < 8 can't access characters by index. IE 8 can only access characters by index on string literals, not string objects.

**\_.templateSettings**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettings) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1074) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array) [Ⓝ](https://www.npmjs.com/package/lodash.templatesettings)

(Object): By default, the template delimiters used by lodash are like those in embedded Ruby (ERB). Change the following template settings to use alternative delimiters.

**\_.templateSettings.escape**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsescape) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1082) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(RegExp): Used to detect data property values to be HTML-escaped.

**\_.templateSettings.evaluate**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsevaluate) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1090) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(RegExp): Used to detect code to be evaluated.

**\_.templateSettings.imports**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsimports) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1114) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(Object): Used to import variables into the compiled template.

**\_.templateSettings.interpolate**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsinterpolate) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1098) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(RegExp): Used to detect data property values to inject.

**\_.templateSettings.variable**

[#](https://github.com/lodash/lodash/tree/3.10.1/doc#_templatesettingsvariable) [Ⓢ](https://github.com/lodash/lodash/blob/3.10.1/lodash.src.js#L1106) [Ⓣ](https://github.com/lodash/lodash/tree/3.10.1/doc#array)

(string): Used to reference the data object in the template text.