CONTENTS Close

example

example

example

example

example

ta from an open text file into a cell array, `C`. The text file
pen to open the file and obtain the `fileID` value. When
alling `fclose(fileID)`.

o `formatSpec`, which is a string of conversion
ughout the entire file and stops when it cannot match

example

file data using the `formatSpec` `N` times, where `N` is a
e file after `N` cycles, call `textscan` again using the
le by calling `textscan` with the same file identifier
s reading at the point where it terminated the last read.

example

om a string, `str`, into cell array `C`. For strings, repeated
nning each time. To restart a scan from the last position,

g, `str`, to `formatSpec`, which is a string of conversion

g data using the `formatSpec` `N` times, where `N` is a

example

tions using one or more `Name,Value` pair arguments, in
evious syntaxes.

example

le or string position at the end of the scan as the second
nts in the previous syntaxes. For a file, this is the value
`extscan`. For a string, `position` indicates how many

collapse all

match each field in `str` to a double-precision floating-point number.

o one decimal digit.

3 digits and a precision of 1. `textscan` reads a total of 3 digits,
er the decimal point. The specifier, `%*1d`, tells `textscan` to skip the

at contains data in the following form:

```
9 inf Nan Yes 5.1+3i
inf  0.001 No 2.2-.5i
10  100   No 3.1+.1i
```

appropriate conversion specifier.

```
%u %f %f %s %f');
```

ells in C.

```
. single]    [3x1 int8]    [3x1 uint32]

 {3x1 cell}    [3x1 double]
```

:{5} is of data type uint32, so the first two elements of C{5} are the
r, or intmax('uint32').

e second column of the data from the previous example.

ıt.

```
2 %d8 %u %f %f %s %f');
```

l in C.

ıow of data type int32.

s example into a cell array, skipping the rest of the line.

```
');
```

ɔn

file, data.csv, that contains

```
3 %f',...
```

extscan function converts the empty value in `C{4}` to `-Inf`, where
t. Because MATLAB represents unsigned integer `-Inf` as `0`, `textscan`
t `-Inf`.

ments

file, `data2.csv`, that contains the lines

at as comments or empty values.

```
','Delimiter',',',...
Style','//');
```

ıat contains

imiter, use the `MultipleDelimsAsOne` parameter, and set the value to

```
)elimiter',',...
```

Collect Numeric Data

that contains:

```
est3
77.3
91.0
92.5
84.6
```

's' four times.

```
,N,'Delimiter','|');
```

```
%f')
```

```
:1 double]    [4x1 double]
```

lse), so textscan returns each column of the numeric data in a

of the file.

```
```

true) to collect the consecutive columns of the same class into a
to indicate that the %f conversion specifier should appear three
repeats many times.

```
limiter','|');

mat('%f',[1,3])],'CollectOutput',1)
```

ected into a single 4-by-3 array.

```
```

elds

text file. Skip a column of strings and a column of integer data.

file called `names.txt` that contains:

file. Use the conversion specifier, `%q` to read a string enclosed by
d string, `%*d` skips an integer field, and `%f` reads a floating-point
ıe `'Delimiter'` name-value pair argument.

```
,'Delimiter',',');
```

le quotation marks enclosing the strings are removed.

contains comma-separated values. The first column of values
d third columns are numeric values.

```
,'ISO-8859-15');
, 100.5 \n');
5, 102.7 \n');
99.8 \n');
```

scheme associated with the file as the last input to `fopen`.

```
,'ISO-8859-15');
```

n the file using the `%{dd % MMMM yyyy}D` specifier. Specify the locale
e pair argument.

```
}D %f %f',...
',',');
```

ses depending on your system locale.

equences in your data.

acter, `\f`. Then, to read the string using `textscan`, call `sprintf` to

```
\fin\fthe\fdead\fof\fnight');
,sprintf('\f'));
```

er than the beginning.

ı reads from the beginning of the string each time. To resume a scan
wo-output argument syntax in your initial call to `textscan`.

ad the first word of the string, and then resume the scan.

```
ead of night';
)c',1);
),'%s');
```

collapse all

a number. Before reading a file with `textscan`, you must use `fopen` to

of one or more conversion specifiers. When `textscan` reads a file or
`matSpec` string. If `textscan` fails to match a data field, it stops reading

ıes the number of cells in output array, `C`.

s for numeric inputs.

	Output Class
	int32
	int8
	int16
	int32
	int64
	uint32
	uint8
	uint16
	uint32
	uint64
	double
	single
	double
	double

s for inputs that include nonnumeric characters.

Details
Read any single character, including a delimiter.
Read a string.
Read a string. If the string begins with a double quotation mark ("), omit the leading quotation mark and its accompanying closing mark, which is the second instance of a lone double quotation mark. Replace escaped double quotation marks (for example, ""abc"") with lone double quotation marks ("abc"). %q ignores any double quotation marks that appear after the closing double quotation mark. Example: '%q' reads '"Joe ""Lightning"" Smith, Jr."' as 'Joe "Lightning" Smith, Jr.'.
Read a string in the same way as %q above, and then convert to a datetime value.
Read a string in the same way as %q above, and then convert to a datetime value. *fmt* describes the format of the input string. The *fmt* input is a string of letter identifiers that is a valid value for the Format property of a datetime. textscan converts strings that do not match this format to NaT values. For more information about datetime display formats, see the Format property for datetime arrays. Example: '%{dd-MMM-yyyy}D' specifies the format of a date such as '01-Jan-2014' .
Read a string in the same way as %q, and then convert to a category name in a categorical array. textscan converts the string, <undefined> to an undefined value in the output categorical array.

	Read only the characters inside the brackets up to the first nonmatching character. To include `]` in the set, specify it first: `%[]...]`. Example: `%[mus]` reads `'summer '` as `'summ'`.
	Exclude characters inside the brackets, reading until the first matching character. To exclude `]`, specify it first: `%[^]...]`. Example: `%[^xrg]` reads `'summer '` as `'summe'`.

ude optional operators, which appear in the following order (includes

in sequence, unless you tell it to ignore a particular field or a portion

rcent character (%) to skip a field or a portion of a character field.

s any conversion specifier identifying the field to skip. `textscan` does put cell for any such fields. `s %s %s %*s %*s %s'` (spaces are optional) converts the string `ging in the dead of night'` to four output cells with the strings `n' 'the' 'night'`
acters, where *n* is an integer less than or equal to the number of field. `%s'` converts `'abcdefg'` to `'defg'`. When the delimiter is a comma, er converts `'abcde,fghijkl'` to a cell array containing `'de';'ijkl'`.
, including delimiter characters.

or digits specified by the field width or precision, or up to the first l point, sign (+ or -), exponent character, and digits in the numeric digits within the field width. For complex numbers, the field width refers d the imaginary part. For the imaginary part, the field width includes + by inserting a number after the percent character (%) in the conversion

e'.

single characters (`%c`), `textscan` also reads delimiter, white-space,

ıg white-space, so `'Day and night'` reads as `'Day and'`.

;f64), you can specify the number of decimal digits to read.

.45.

ɔ to the `formatSpec` string.

ed as a positive integer.

`int32` | `int64` | `uint8` | `uint16` | `uint32` | `uint64`

e,Value arguments. `Name` is the argument name and `Value` is the
single quotes (`' '`). You can specify several name and value pair
NameN,ValueN.

,'HeaderLines',3,'Delimiter',',') skips the first three lines of the
ıg commas as a delimiter.

ermining data concatenation

tion, specified as the comma-separated pair consisting of
f `true`, then `textscan` concatenates consecutive output cells of the
e array.

ext to ignore

as the comma-separated pair consisting of `'CommentStyle'` and a

ɡnore characters following the string on the same line. Specify a cell
› ignore characters between the strings.

art of each field, not within a field.

ıma-separated pair consisting of `'DateLocale'` and a string in the
-1 two-letter code that specifies a language, and *YY* is an uppercase
ntry. For a list of common values for the locale, see the `Locale` name-
ı.

ı textscan should interpret month and day of week names and
ıg the %D format specifier.

mma-separated pair consisting of `'Delimiter'` and a string or a cell
a cell array of strings.

cters as separate delimiters, and returns an empty value to the output

ıiter is white-space. White-space can be any combination of space (`'`
s. If you do not specify a delimiter, then:

ıe white-space characters. The default white-space characters are `'`
name-value pair argument to specify alternate white-space

e characters as a single delimiter.

e sequences as a delimiter, `textscan` converts that sequence to the

y numeric fields

limited text files, specified as the comma-separated pair consisting of

ıa-separated pair consisting of `'EndOfLine'` and a string. The string
The default end-of-line sequence depends on the format of your file
ı carriage return (`'\r'`), or a combination of the two (`'\r\n'`).

sequence at the end of the last line in a file, then `textscan` returns
at individual cells in output cell array, `C`, are the same size.

-separated pair consisting of `'ExpChars'` and a string. The default

ma-separated pair consisting of `'HeaderLines'` and a positive
uding the remainder of the current line.

iter handling

omma-separated pair consisting of `'MultipleDelimsAsOne'` and
consecutive delimiters as a single delimiter. Repeated delimiters
a single delimiter. You must also specify the `Delimiter` option.

can fails to read or convert

ert, specified as the comma-separated pair consisting of
f `true`, `textscan` terminates without an error and returns all fields
rror and does not return an output cell array.

ty value

he comma-separated pair consisting of `'TreatAsEmpty'` and a single
applies to numeric fields.

ma-separated pair consisting of `'Whitespace'` and a string of one or
racter, `char(32)`, to any specified `Whitespace`, unless `Whitespace` is
ng conversion specifier.

e sequences as any white-space character, `textscan` converts that
cter:

collapse all

matSpec, the textscan function returns a K-by-1 MATLAB numeric
e number of times that textscan finds a field matching the specifier.

tSpec, the textscan function returns a K-by-1 cell vector of strings,
finds a field matching the specifier. For each character conversion
returns a K-by-M character array, where M is the field width.

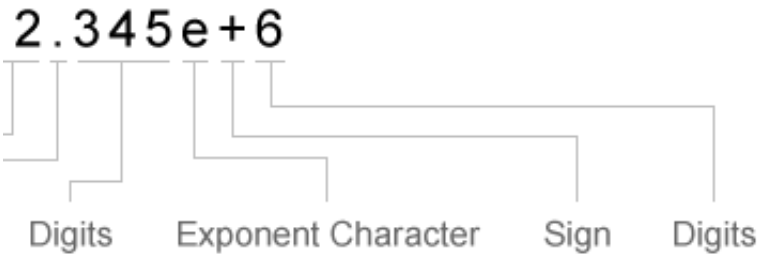
specifier in formatSpec, the textscan function returns a K-by-1
ll array, C, where K is the number of times that textscan finds a field

eturned as an integer of class double. For a file, ftell(fileID)
tscan. For a string, position indicates how many characters

collapse all

fied output type according to MATLAB rules regarding overflow,
For example, MATLAB represents an integer NaN as zero. If textscan
r format specifier (such as %d or %u), it returns the empty value as zero

ecifier, textscan reads until it finds a delimiter or an end-of-line
conversion specifier, textscan reads until it finds a nonnumeric
ch the data to a particular conversion specifier, it attempts to match
e formatSpec string. Sign (+ or -), exponent characters, and decimal



	Digits	Exponent Character	Sign	Digits
nt	If there is a decimal point, read one or more digits that immediately	Read one exponent character if it exists.	If there is an exponent character, read one sign character.	If there is an exponent character, read one or more digits that follow it.

	follow it.			

whole into a complex numeric field, converting the real and imaginary
d or %f). Valid forms for a complex number are:

	Example: `5.7-3.1i`
	Example: `-7j`

omplex number. `textscan` interprets embedded white space as a field

able | uiimport | xlsread