

LoadWave

Keyword	Information Following Keyword
HEIGHT	Height of the picture in pixels.
PHYSWIDTH	Physical width of the picture in points.
PHYSHEIGHT	Physical height of the picture in points.

See Also

See **Pictures** on page III-509 for general information on how Igor handles pictures.

The **ImageLoad** operation for loading PICT and other image file types into waves, and the **PICTInfo** function.

LoadWave

LoadWave [*flags*] [*fileNameStr*]

The LoadWave operation loads data from the named Igor binary, Igor text, delimited text, fixed field text, or general text file into waves. LoadWave can load 1D and 2D data from delimited text, fixed field text and general text files, or data of any dimensionality from Igor binary and Igor text files.

Parameters

The file to be loaded is specified by *fileNameStr* and */P=pathName* where *pathName* is the name of an Igor symbolic path. *fileNameStr* can be a full path to the file, in which case */P* is not needed, a partial path relative to the folder associated with *pathName*, or the name of a file in the folder associated with *pathName*. If LoadWave can not determine the location of the file from *fileNameStr* and *pathName*, it displays a dialog allowing you to specify the file.

If you use a full or partial path for *fileNameStr*, see **Path Separators** on page III-451 for details on forming the path.

If *fileNameStr* is "Clipboard" and */P=pathName* is omitted, LoadWave takes its data from the Clipboard rather than from a file. This is not implemented for binary loads.

If *fileNameStr* is omitted or is "", or if the */I* flag is used, LoadWave presents an Open File dialog from which you can choose the file to load.

Flags

<i>/A</i>	<p>"Auto-name and go" option (used with <i>/G</i>, <i>/F</i> or <i>/J</i>).</p> <p>This skips the dialog in which you normally enter wave names. Instead it automatically assigns names of the form <i>wave0</i>, <i>wave1</i>, choosing names that are not already in use. When used with <i>/W</i>, it reads wave names from the file instead of automatically assigning names and <i>/A</i> just skips the wave name dialog. The <i>/B</i> flag can also override names specified by <i>/A</i>.</p> <p>See LoadWave Generation of Wave Names on page II-142 for further discussion.</p>
<i>/A=baseName</i>	Same as <i>/A</i> but it automatically assigns wave names of the form <i>baseName0</i> , <i>baseName1</i> .
<i>/B=columnInfoStr</i>	Specifies the name, format, numeric type, and field width for columns in the file. See Specifying Characteristics of Individual Columns on page II-145 for details.
<i>/C</i>	This is used in experiment recreation commands generated by Igor to force experiment recreation to continue if an error occurs in loading a wave.
<i>/D</i>	Creates double precision waves. (Used with <i>/G</i> , <i>/F</i> or <i>/J</i> .) The <i>/B</i> flag can override the numeric precision specified by the <i>/D</i> flag.
<i>/E=editCmd</i>	Controls table creation: <div><i>editCmd</i>=1: Makes a new table containing the loaded waves. <i>editCmd</i>=2: Appends the loaded waves to the top table. If no table exists, a new table is created. <i>editCmd</i>=0: Same as if <i>/E</i> had not been specified (loaded waves are not put in any table).</div>

/ENCG=textEncoding

/ENCG={textEncoding, tecOptions}

Specifies the text encoding of the plain text file being loaded.

This flag was added in Igor Pro 7.00.

This flag is ignored when loading an Igor binary wave file.

See **Text Encoding Names and Codes** on page III-490 for a list of accepted values for *textEncoding*.

For most purposes the default value for *tecOptions*, 3, is fine and you can use */ENCG=textEncoding* instead of */ENCG={textEncoding, tecOptions}*.

tecOptions is an optional bitwise parameter that controls text encoding conversion. See **Setting Bit Parameters** on page IV-12 for details about bit settings.

tecOptions is defined as follows:

- Bit 0: If cleared, the presence of null bytes causes LoadWave to consider the text invalid in all byte-oriented text encodings. This makes it easier for LoadWave to identify UTF-16 and UTF-32 text which usually contains null bytes.
If set (default), null bytes are allowed in byte-oriented text encodings.
- Bit 1: If cleared LoadWave does not validate text if the specified text encoding is UTF-8.
If set (default), LoadWave validates text even if the text encoding is UTF-8.
- Bit 2: If cleared (default) LoadWave validates the text in the specified text encoding except that validation for UTF-8 is skipped if bit 1 is cleared.
If set, LoadWave assumes that the text is valid in the specified text encoding and does not validate it. Setting this bit makes LoadWave slightly faster but it is usually inconsequential.
- Bit 3: If cleared (default), LoadWave presents the Choose Text Encoding dialog if the specified text encoding is not valid for the text in the file.
If set, LoadWave does not present the Choose Text Encoding dialog if the text is not valid in the specified text encoding and instead returns an error. Set this bit if you are running an automated procedure and do not want it interrupted by a dialog.

See **LoadWave Text Encoding Issues** on page II-149 for further discussion.

/F={numColumns, defaultFieldWidth, flags}

Indicates that the file uses the fixed field file format. Most FORTRAN programs generate files in this format.

numColumns is the number of columns of data in the file.

defaultFieldWidth is the default number of bytes in each column. If the columns do not all have the same number of bytes, you need to use the */B* flag to provide more information to LoadWave.

flags is a bitwise parameter that controls the conversion of text to values. The bits are defined as follows:

- Bit 0: If set, any field that consists entirely of the digit "9" is taken to be blank. A field that consists entirely of the digit "9" except for a leading "+" or "-" is also taken to be blank.

All other bits are reserved and must be cleared.

/G

Indicates that the file uses the general text format.

LoadWave

/H	<p>Loads the wave into the current experiment and severs the connection between the wave and the file. When the experiment is saved, the wave copy will be saved as part of the experiment. For a packed experiment this means it is saved in the packed experiment file. For an unpacked experiment this means it is saved in the experiment's home folder.</p> <p>See Sharing Versus Copying Igor Binary Wave Files on page II-156.</p>
/I	<p>Forces LoadWave to display an Open File dialog even if the file is fully specified via /P and <i>fileNameStr</i>.</p>
/J	<p>Indicates that the file uses the delimited text format.</p>
/K= <i>k</i>	<p>Controls how to determine whether a column in the file is numeric or text (only for delimited text and fixed field text files).</p> <p><i>k</i>=0: Deduces the nature of the column automatically.</p> <p><i>k</i>=1: Treats all columns as numeric.</p> <p><i>k</i>=2: Treats all columns as text.</p> <p>This flag as well as the ability to load text data into text waves were added in Igor Pro 3.0. The default for the LoadWave operation is /K=1, meaning that it will treat all columns as numeric. We did this so that existing procedures would behave the same in Igor Pro 3.0 as before. Use /K=0 when you want to load text columns into text waves. /K=2 may have use in a text-processing application.</p> <p>For finer control, the /B flag specifies the format of each column in the file individually.</p>
/L={ <i>nameLine</i> , <i>firstLine</i> , <i>numLines</i> , <i>firstColumn</i> , <i>numColumns</i> }	<p>Affects loading delimited text, fixed field text, and general text files only (/J, /F or /G). /L is accepted no matter what the load type but is ignored for Igor binary and Igor text loads. Line and column numbers start from 0.</p> <p><i>nameLine</i> is the number of the line containing column names. For general text loads, 0 means auto. See Loading General Text Files on page II-138 for details.</p> <p><i>firstLine</i> is the number of the first line to load into a wave. For general text loads, 0 means auto. See Loading General Text Files on page II-138 for details.</p> <p><i>numLines</i> is the number of lines that should be treated as data. 0 means auto which loads until the end of the file or until the end of the block of data in general text files. The <i>numLines</i> parameter can also be used to make loading very large files more efficient. See Loading Very Large Files on page II-150 for details.</p> <p><i>firstColumn</i> is the number of the first column to load into a wave. This is useful for skipping columns.</p> <p><i>numColumns</i> is the number of columns to load into a wave. 0 means auto, which loads all columns.</p>
/M	<p>Loads data as matrix wave. If /M is used then it ignores the /W flag (read wave names) and follows the /U flags instead.</p> <p>The wave is autonamed unless you provide a specific wave name using the /B flag.</p> <p>The type of the wave (numeric or text) is determined by an assessment of the type of the first loaded column unless you override this using the /K flag or the /B flag.</p> <p>See The Load Waves Dialog for Delimited Text — 2D on page II-133 for further information.</p>
/N	<p>Same as /A except that, instead of choosing names that are not in use, it overwrites existing waves.</p> <p>See LoadWave Generation of Wave Names on page II-142 for further discussion.</p>

<i>/N=baseName</i>	Same as <i>/N</i> except that it automatically assigns wave names of the form <i>baseName0</i> , <i>baseName1</i> .
--------------------	---

<i>/NAME={namePrefix, nameSuffix, nameOptions}</i>	<p>The <i>/NAME</i> flag overrides or combines with the other flags (<i>/A</i>, <i>/N</i>, <i>/W</i>, and <i>/B</i>) that affect the names of waves loaded by LoadWave.</p> <p><i>/NAME</i> was added in Igor Pro 9.00. It applies to Load General Text (<i>/G</i>), Load Delimited Text (<i>/J</i>), and Load Fixed Field Text (<i>/F</i>), not when loading Igor binary files or Igor Text files.</p> <p>The <i>/NAME</i> flag is primarily intended to provide an easy way to incorporate the file name in the wave names. For example, if you are loading a single wave from a file named File.txt, you can create a single wave named File and, if you are loading three waves from a file named File.txt, you can create three waves named File0, File1, and File2.</p> <p><i>namePrefix</i> is "" or text to be prepended to the final wave names.</p> <p><i>nameSuffix</i> is "" or text to be appended to the final wave names.</p> <p>If both <i>namePrefix</i> and <i>nameSuffix</i> are "", it is as if the <i>/NAME</i> flag was omitted altogether.</p> <p>In the construction of the final wave names, LoadWave replaces ":filename:" in <i>namePrefix</i> or <i>nameSuffix</i> with the name of the file being loaded without the file name extension.</p> <p><i>nameOptions</i> is a bitwise parameter. The bits are defined as follows:</p> <ul style="list-style-type: none"> Bit 0: Include normal name. If set, the "normal" wave name (the name that would be used if <i>/NAME</i> were omitted) is included in the final name between <i>namePrefix</i> and <i>nameSuffix</i>. Bit 1: Include normal name. If set, the "normal" wave name (the name that would be used if <i>/NAME</i> were omitted) is included in the final name between <i>namePrefix</i> and <i>nameSuffix</i>. Bit 2: Include suffix number when loading multiple waves. If multiple waves, a suffix number is appended unless suppressed by bit 3. Bit 3: Include suffix number only if there is a name conflict. If set and there are no name conflicts, this overrides bits 1 and 2 and prevents appending suffix numbers. Bit 4: Choose suffix number to create unique names. If set, LoadWave chooses suffix numbers to avoid conflicts with existing waves. If cleared suffix numbers start from 0 and increment by one for each loaded wave. Bit 5: Allow liberal names (not recommended). <p>See Setting Bit Parameters on page IV-12 for details about bit settings.</p> <p>See LoadWave Generation of Wave Names on page II-142 for further discussion.</p>
--	---

<i>/O</i>	Overwrite existing waves in case of a name conflict.
-----------	--

<i>/P=pathName</i>	Specifies the folder to look in for <i>fileNameStr</i> . <i>pathName</i> is the name of an existing symbolic path.
--------------------	--

<i>/Q</i>	Suppresses the normal messages in the history area.
-----------	---

<i>/R={languageName, yearFormat, monthFormat, dayOfMonthFormat, dayOfWeekFormat, layoutStr, pivotYear}</i>	Specifies a custom date format for dates in the file. If the <i>/R</i> flag is used, it overrides the date setting that is part of the <i>/V</i> flag.
--	--

languageName controls the language used for the alphabetic date components, namely the month and the day-of-week. *languageName* is one of the following:

Default

Chinese	ChineseSimplified	Danish	Dutch
English	Finnish	French	German
Italian	Japanese	Korean	Norwegian
Portuguese	Russian	Spanish	Swedish

Default means the system language on Macintosh or the user default language on Windows.

yearFormat is one of the following codes:

- 1: Two digit year.
- 2: Four digit year.

monthFormat is one of the following codes:

- 1: Numeric, no leading zero.
- 2: Numeric with leading zero.
- 3: Abbreviated alphabetic (e.g., Jan).
- 4: Full alphabetic (e.g., January).

dayOfMonthFormat is one of the following codes:

- 1: Numeric, no leading zero.
- 2: Numeric with leading zero.

dayOfWeekFormat is one of the following codes:

- 1: Abbreviated alphabetic (e.g., Mon).
- 2: Full alphabetic (e.g., Monday).

layoutStr describes which components appear in the date in what order and what separators are used. *layoutStr* is constructed as follows (but with no line break):

```
"<component keyword><separator>  
<component keyword><separator>  
<component keyword><separator>  
<component keyword>"
```

where <component keyword> is one of the following:

Year
Month
DayOfMonth
DayOfWeek

and <separator> is a string of zero to 15 bytes.

Starting from the end, parts of the layout string must be omitted if they are not used.

Extraneous spaces are not allowed in *layoutStr*. Each separator must be no longer than 15 bytes. No component can be used more than once. Some components may be used zero times.

pivotYear determines how LoadWave interprets two-digit years. If the year is specified using two digits, *yy*, and is less than *pivotYear* then the date is assumed to be 20*yy*. If the two digit year is greater than or equal to *pivotYear* then the year is assumed to be 19*yy*. *pivotYear* must be between 4 and 40.

See **Loading Custom Date Formats** on page II-144 for further discussion of date formats.

/T

Indicates that the file uses the Igor text format.

Although LoadWave is generally thread-safe, it is not thread-safe to load an Igor text file containing an Igor command (e.g., "X <command>").

/U={*readRowLabels*, *rowPositionAction*, *readColLabels*, *colPositionAction*}

These parameters affect loading a matrix (/M) from a delimited text (/J) or a fixed field text (/F) file. They are accepted no matter what the load type is but are ignored when they don't apply.

If *readRowLabels* is nonzero, it reads the first column of data in the file as the row labels for the matrix wave.

rowPositionAction has one of the following values:

- 0: The file has no row position column.
- 1: Uses the row position column to set the row scaling of the matrix wave.
- 2: Creates a 1D wave containing the values in the row position column. The name of the 1D wave will be the same as the matrix wave but with the prefix "RP_".

The *readColLabels* and *colPositionAction* parameters have analogous meanings. The prefix used for the column position wave is "CP_".

See Chapter II-9, **Importing and Exporting Data** for further details.

/V={*delimsStr*, *skipCharsStr*, *numConversionFlags*, *loadFlags*}

These parameters affect loading delimited text (/J) and fixed field text (/F) data and column names. They do not affect loading general text (/G). They are accepted no matter what the load type is but are ignored when they don't apply. These parameters should rarely be needed.

delimsStr is a string expression containing the characters that should act as delimiters for delimited file loads. (When loading data as general text, the delimiters are always tab, comma and space.) The default is "\t," for tab and comma. You can specify the space character as a delimiter, but it is always given the lowest priority behind any other delimiters contained in *delimsStr*. The low priority means that if a line of text contains any other delimiter besides the space character, then that delimiter is used rather than the space character.

skipCharsStr is a string expression containing characters that should always be treated as garbage and skipped when they appear before a number. The default is "\$" for space and dollar sign. This parameter should rarely be needed.

numConversionFlags is a bitwise parameter that controls the conversion of text to numbers. The bits are defined as follows:

- Bit 0: If set: dates are dd/mm/yy.
If cleared: dates are mm/dd/yy.
- Bit 1: If set: decimal character is comma.
If cleared: it is period.
- Bit 2: If set: thousands separators in numbers are ignored when loading delimited text only (LoadWave/J).

The thousands separator is the comma in 1,234 or, if comma is the decimal character, the dot in 1.234.

Most numeric data files do not use thousands separators and searching for them slows loading down so this bit should usually be 0.

This bit has no effect if the thousands separator (e.g., comma) is also a delimiter character as specified by *delimsStr*.

All other bits are reserved and must be cleared.

See **Setting Bit Parameters** on page IV-12 for details about bit settings.

If the /R flag is used to specify the date format, this overrides the setting of bit 0.

loadFlags is a bitwise parameter that controls the overall load. The bits are defined as follows:

- Bit 0: If set, ignores blanks at the end of a column. This is appropriate if the file contains columns of unequal length. Set *loadFlags* = 1 to set bit 0.
- Bit 1: If set, when the /W flag is specified and the line containing column labels starts with one or more space characters, the spaces are taken to be a blank column name. The resulting column will be named Blank. Use this if both the line containing column labels and the lines containing data start with leading spaces in a space-delimited file. Set *loadFlags* = 2 to set bit 1.
- Bit 2: If set: Disables pre-counting of lines of data. See **Loading Very Large Files** on page II-150.
- Bit 3: If set: Disables unescaping of backslash characters in text columns. See **Escape Sequences** on page II-150 below.
- Bit 4: If set: Disables support for quoted strings.

Setting this bit improves compatibility with Igor7. It applies to delimited text only and should rarely be needed. See **Quoted Strings in Delimited Text Files** on page II-135 and **Delimited Text Compatibility With Igor7** on page II-136 for background information.

/W Looks for wave names in a file. (With /G, /F, and /J.)

LoadWave cleans up column names to create standard, not liberal, wave names. See **Object Names** on page III-501 and **CleanupName** for details.

Use /W/A to read wave names from the file and then continue the load without displaying the normal wave name dialog.

See **LoadWave Generation of Wave Names** on page II-142 for further discussion.

Details

Without the /G, /F, /J, or /T flags, LoadWave loads Igor binary wave files.

If you use /P=*pathName*, note that it is the name of an Igor symbolic path, created via **NewPath**. It is not a file system path like "hd:Folder1:" or "C:\\Folder1\\". See **Symbolic Paths** on page II-22 for details.

When loading a general text file, the delimiters are always tab, comma and space.