

# Technotes



## QuickTime Teletext Component APIs

---

**Technote 1010****OCTOBER 1995**

---

This Technical Note describes the Teletext component APIs.

The Teletext component is included in the Video Startup extension which is part of the Apple Video Player software. The component provides a standard API for accessing the Teletext hardware available on the international and UK tuner cards.

See [Inside Macintosh: QuickTime](#) and [Inside Macintosh: QuickTime Components](#) for further documentation.

**Note:**

This document assumes you have used Teletext before and understand the basic functionality from a user point of view.

**Contents**

- [About Teletext Component APIs](#)
- [Using Teletext Component APIs](#)
- [TeleText Component APIs](#)

## About Teletext Component APIs

The Apple international and UK tuner cards include Teletext decoding hardware. The hardware also includes an Teletext page cache (currently 512K, but this will change in the future). The Teletext decoder hardware automatically decodes each Teletext page as it is received by the TV tuner and places it in the page cache. The page cache can store up to 512 of the most recently received pages. Because most Teletext systems broadcast fewer than 512 pages, the page cache contains almost every available Teletext page. This provides immediate access to any page on the system.

### Teletext Page Data

The Teletext page data consists of 25 lines of data, each containing 40 characters. Each character may be a visible text character, a graphics character, or a special control character. Control characters are used to set text color, text size, text or graphics mode, etc. Interpretation of the control codes is not terribly complex, but is beyond the scope of this document.

The first eight characters of the first line do not contain any visible data. This area is used by some televisions to display the current page number. The last eight characters of the first line contain the current time.

The last line is used to display Fastext information. If no Fastext information is available for the current page, the last line will contain spaces.

Visible text characters are not encoded using standard Macintosh character encoding. The data may be displayed as-is using the special Teletext font, but when displaying it using any other font or when saving it to disk or the clipboard, the data must be converted to standard Macintosh ASCII encoding.

Each page also contains a small amount of other information: Fastext page numbers if available, a local index page number if available, and the character set the page should be displayed with (i.e., English, German, etc.).

## Using Teletext Component APIs

### Accessing a Teletext Page

The general algorithm for accessing the Teletext pages is as follows:

- Issue a request for the required page.
- Periodically check to see if the page is available. Pages in the page cache will be available almost immediately. Other pages won't be available until they have been received.
- Once the page is available, read the page data into a buffer. The page can then be displayed, saved to disk, etc.

If the page is being displayed to the user, the application must periodically check to see if the page has been updated. "Updated" simply means that the page has been retransmitted by the broadcast station since the last time the page data was read, the actual contents of the page may or may not have changed. If the page has been updated, the application should read the new page data, check to see if the data has actually changed, and redisplay the page if necessary.

If the page is being displayed to the user, the application should also update the time field (i.e., the last 8 characters of the first line) once every second.

### Using the Teletext component

All access to the Teletext information is handled by the Teletext component. The following code demonstrates how to find and open the Teletext component:

```
ComponentDescription theDesc;

ComponentID;

ComponentInstance teletextInst;

theDesc.componentType = 'tltx';

theDesc.componentSubType = 0;

theDesc.componentManufacturer = 'appl';

theDesc.componentFlags = 0L;

theDesc.componentFlagsMask = 0L;

id = FindNextComponent(nil, &theDesc);

if (id != nil)

    teletextInst = OpenComponent(id);
```

`FindNextComponent` will return `nil` if the Teletext component does not exist or if the component has determined that the Teletext hardware does not exist. `OpenComponent` will return `nil` if the Teletext component has already been opened by another application.

When the application is finished using the component, it should close the component using the `CloseComponent` call.

## Teletext Component APIs

### TTGetFastextPages

```
pascal ComponentResult TTGetFastextPages(ComponentInstance ci, short *pages);
```

ci            component instance

pages        pointer to an array of kTTFastextPages (4) page numbers

Returns `ttPageNotAvailErr` if requested page is not yet in memory. Returns `ttNoPageLinksErr` if Fastext information is not available for this channel.

This function returns the page numbers associated with the four colored Fastext links. Page number 0 is for the red link, page number 1 is for the green link, page number 2 is for the yellow link, and page number 3 is for the cyan link. There may be fewer than four Fastext items available for a given page. A page number of zero indicates that the Fastext link is not available for the current page.

## TTGetLocalIndexPage

```
pascal ComponentResult TTGetLocalIndexPage(ComponentInstance ci, short *page);
```

ci            component instance

page         pointer to a page number

Returns `ttPageNotAvailErr` if requested page is not yet in memory. Returns `ttNoPageLinksErr` if there is no local index page indicated for this page.

This function returns the local index page for the current page. Some stations specify a local index page for each Teletext page transmitted. For example, all of the pages in the sports section might specify the sports index page as the local index page. In turn, the sports index page might specify the news index page as its local index page. The local index page is generally associated with the "Index" or "i" button on a remote control. If no local index page is specified for a page, use page 100 as the index page.

## TTGetPageFlags

```
pascal ComponentResult TTGetPageFlags(ComponentInstance ci, long *flags);
```

ci            component instance

flags        pointer to a 32-bit flag word

Returns `ttPageNotAvailErr` if requested page is not yet in memory.

This function returns various flags associated with the current page. Each of the flags is defined below:

`kTTLanguageFlags` = A 3-bit language value that specifies with character set to use when displaying the current page. The defined values are shown below. Undefined values should be treated as English.

```
kTTLangEnglish = 0
```

```
kTTLangFrench = 1
```

```
kTTLangSwedish = 2
```

```
kTTLangGerman = 4
```

```
kTTLangSpanish = 5
```

```
kTTLangItalian = 6
```

`kTTSubtitleFlag` = True if this page is a subtitle page. Subtitles are similar to US closed captioning. They are displayed in a box overlaying the video. Subtitles are generally transmitted on page 888.

`kTTNewsFlashFlag` = True if this page is a newsflash page. Newsflashes are also displayed in a box overlaying the video, but may be transmitted on any page.

## TTGetPageNumber

```
pascal ComponentResult TTGetPageNumber(ComponentInstance ci, short *page, short *subpage);
```

ci            component instance

page        pointer to a page number

subpage    pointer to a subpage number

This function returns the page and subpage numbers last requested using the `TTRequestPage` function.

## TTIsBroadcasting

```
pascal ComponentResult TTIsBroadcasting(ComponentInstance ci, Boolean *flag);
```

ci            component instance

flag        pointer to a Boolean flag

This function returns `true` if Teletext information is being broadcast on the current channel.

## TTIsPageAvailable

```
pascal ComponentResult TTIsPageAvailable(ComponentInstance ci, Boolean *available);
```

ci            component instance

available    pointer to a Boolean flag

This function returns `true` if the page requested using `TTRequestPage` has been received and placed in memory.

## TTIsPageUpdated

```
pascal ComponentResult TTIsPageUpdated(ComponentInstance ci, Boolean *updated);
```

ci            component instance

updated    pointer to Boolean flag

Returns `ttPageNotAvailErr` if requested page is not yet in memory.

This function returns `true` if current page has been updated since the last call to `TTReadPage`. "Updated" simply means that a new copy of the page has been received from the broadcast station, the contents of the page may or may not have changed.

## TTReadPage

```
pascal ComponentResult TTReadPage(ComponentInstance ci, Ptr buffer);
```

ci            component instance

buffer    pointer to a 1000 byte page buffer

Returns `ttPageNotAvailErr` if requested page is not yet in memory. Returns `ttParamErr` if the buffer pointer is `nil`.

This function reads an entire Teletext page into the specified buffer. The page must first be requested using the `TTRequestPage` function. The page is organized into 25 lines with 40 characters per line. See the Teletext specification for information on decoding the page data.

## TTReadLine

```
pascal ComponentResult TTReadLine(ComponentInstance ci, Ptr buffer,
                                   short line, short startcol, short length);
```

ci            component instance

buffer    pointer to an appropriately sized buffer

line        line number (0-24)

startcol   starting column number (0-39)

length     number of bytes to read (1-40)

Returns `ttPageNotAvailErr` if requested page is not yet in memory. Returns `ttParamErr` if the buffer pointer is nil, or if any of the other parameters is out of range.

This function reads a portion of a Teletext line and places it into the specified buffer.

## TTRequestPage

```
pascal ComponentResult TTRequestPage(ComponentInstance ci, short page, short subpage);
```

ci        component instance

page     page number (100-899)

subpage   subpage number (0-99)

Returns `ttPageNotAvailErr` if requested page is not yet in memory. Returns `ttParamErr` if the page or subpage numbers is out of range.

This function tells the Teletext hardware to load the requested page into internal memory. Requesting a page is an asynchronous operation. If the page number is in the external page cache, the page will be loaded into memory almost immediately. If it is not in the external page cache, the page won't be loaded into memory until the page is transmitted again. Use the `TTIsPageAvailable` function to determine if the page has been loaded into memory.

## TTReset

```
pascal ComponentResult TTReset(ComponentInstance ci);
```

ci        component instance

This function resets the Teletext hardware, clearing the external page cache and setting the requested page number to 100. This function should be called whenever the user changes channels.

## Teletext Component Constants

```
#define        ttUnimpErr                        -2201        /* feature unimplemented */
#define        ttParamErr                       -2202        /* bad parameter (out of range, etc) */
#define        ttPageNotAvailErr               -2203        /* requested page is not yet available */
#define        ttTooManyInstErr               -2204        /* too many instances of tuner */
#define        ttNoPageLinksErr               -2205        /* no fasttext or index links available */

#define        kTTFastextPages                4            /* number of Fastext page links */
#define        kTTLinesPerPage                25        /* number of lines per page */
#define        kTTLineLength                  40        /* number of characters per line */
#define        kTTTimeLength                  8           /* number of characters in time field */

#define        kTTLanguageFlags               ((1<<2)+(1<<1)+(1<<0))   /* language type*/
#define        kTTSubtitleFlag                (1<<3)        /* subtitle flag*/
#define        kTTNewsFlashFlag               (1<<4)        /* news flash flag */
```

```
#define          kTTLangEnglish          0          /* language values */
#define          kTTLangFrench           1
#define          kTTLangSwedish          2
#define          kTTLangGerman           4
#define          kTTLangSpanish           5
#define          kTTLangItalian           6

#define          kTTRequestPageSelect     6
#define          kTTReadPageSelect        7
#define          kTTReadLineSelect        8
#define          kTTIsPageAvailableSelect 9
#define          kTTIsPageUpdatedSelect   10
#define          kTTGetPageFlagsSelect     11
#define          kTTGetPageNumberSelect    12
#define          kTTGetFastextPagesSelect  13
#define          kTTResetSelect            14
#define          kTTIsBroadcastingSelect   15
#define          kTTGetLocalIndexPageSelect 16
```

```
pascal ComponentResult TTGetFastextPages(ComponentInstance ac, short *pages)
    ComponentCallNow(kTTGetFastextPagesSelect, sizeof(Ptr));
```

```
pascal ComponentResult TTGetLocalIndexPage(ComponentInstance ac, short *page)
    ComponentCallNow(kTTGetLocalIndexPageSelect, sizeof(Ptr));
```

```
pascal ComponentResult TTGetPageFlags(ComponentInstance ac, long *flags)
    ComponentCallNow(kTTGetPageFlagsSelect, sizeof(Ptr));
```

```
pascal ComponentResult TTGetPageNumber(ComponentInstance ac, short *page, short *subpage)
    ComponentCallNow(kTTGetPageNumberSelect, sizeof(Ptr) + sizeof(Ptr));
```

```
pascal ComponentResult TTIsBroadcasting(ComponentInstance ac, Boolean *flag)
    ComponentCallNow(kTTIsBroadcastingSelect, sizeof(Ptr));
```

```
pascal ComponentResult TTIsPageAvailable(ComponentInstance ac, Boolean *available)
    ComponentCallNow(kTTIsPageAvailableSelect, sizeof(Ptr));
```

```
pascal ComponentResult TTIsPageUpdated(ComponentInstance ac, Boolean *updated)
```

```
ComponentCallNow(kTTIsPageUpdatedSelect, sizeof(Ptr));

pascal ComponentResult TTReadPage(ComponentInstance ac, Ptr buffer)
    ComponentCallNow(kTTReadPageSelect, sizeof(Ptr));

pascal ComponentResult TTReadLine(ComponentInstance ac, Ptr buffer,
                                   short line, short startcol, short length)
    ComponentCallNow(kTTReadLineSelect, sizeof(Ptr) + sizeof(short)*3);

pascal ComponentResult TTRequestPage(ComponentInstance ac, short page, short subpage)
    ComponentCallNow(kTTRequestPageSelect, sizeof(short) + sizeof(short));

pascal ComponentResult TTReset(ComponentInstance ac)
    ComponentCallNow(kTTResetSelect, 0);
```

## Further Reference

- [\*Inside Macintosh: QuickTime\*](#)
- [\*Inside Macintosh: QuickTime Components\*](#)
- [\*QuickTime 2.0 Developer Guide for Macintosh\*](#)