

Comments

Introduction

Commenting source code is important as it makes your code more maintainable. You can return back to a code you typed several month ago and recall and understand what it does quickly if your comments are up to date.

REM lines

Traditional BASIC dialects (e.g. Sinclair BASIC) uses REM to comment lines. REM is an abbreviation of *REMark*. ZX BASIC allows it:

```
10 REM This line has a comment
20 PRINT "Hello World": REM The following sentence is ignored: PRINT "Hello Again"
```

REM exists for the sake of retrocompatibility with old Sinclair BASIC listings.

Apostrophe character


Newer dialects uses the apostrophe character (') as a shortened REM. ZX BASIC also allows it, so the above listing could be also rewritten this way:

```
10 ' This line has a comment
20 PRINT "Hello World": 'The following sentence is ignored: PRINT "Hello Again"
```

Everything beyond the apostrophe char will be ignored. Since line numbers can also be omitted, the above listing could be rewritten as:

```
' This line has a comment
PRINT "Hello World" 'The following sentence is ignored: PRINT "Hello Again"
```

Notice now the missing colon at the end of the **PRINT** statement. REM, like any other BASIC sentence, requires a colon when it is preceded by another one, whilst apostrophe does not.

The apostrophe character was used in Sinclair BASIC as a **PRINT** modifier. But here, in ZX BASIC it is **always** a commenter char.  [v: latest](#) ▼

Multi-line comments

Multi-line comments are marked with the tokens `/*` and `*/`. All text between the two markers is considered comment text and is not compiled.

Multi-line comments can span several lines, and can also be used in the middle of statements. After the end of the comment, the statement will continue to be parsed as normal (even if the comment crosses line breaks).

```
/* Multi-line  
comment */  
  
Print "Hello" /* embedded comment */ " world"
```