

# External Component Version Numbers

As third-party Omnis component suppliers we are in the unfortunate position that we have to support numerous different versions of Omnis Studio for which we have to build and distribute different DLLs. Our support team has been contacted on numerous occasions when the wrong DLLs had been used for a particular version of Omnis Studio causing crashes, runtime errors, or less obvious compatibility issues. Some time ago we took the decision to alter our component version numbers so that one can identify the version of Studio for which a DLL was built.

All our components will be gradually updated as part of standard maintenance. The release notes for each product that has been updated will state so and link to this document.

This document will explain the new format, and how one can identify the DLL to prevent compatibility issues that may circumvent your internal QA. At the end of this document we will list the current supported versions of Studio and how they relate to our new version numbers. Included will also be a list of our products which have been updated to the new format. Consequently, this document will be updated on a regular basis as we release new versions of our products or when Tiger Logic releases new versions of Studio that require new builds.

## IMPORTANT DOCUMENT CHANGES:

We have updated the supported studio versions table for Studio 8.0. Make sure you study this table if you are using this version.

<b>The old format</b>	<b>2</b>
<b>The new format</b>	<b>2</b>
<b>Checking the version programatically</b>	<b>2</b>
<b>Checking the file version of the DLL</b>	<b>3</b>
<b>Supported Studio versions</b>	<b>3</b>
<b>Updated External Components</b>	<b>4</b>

## The old format

All our Studio software now uses the new version number format thus we have removed instructions regarding old version numbers.

## The new format

The new version number format includes the major and minor version of Omnis Studio for which the DLL was built and the four digit component version consisting of major, minor, maintenance and patch digits. The complete version number will be “ab.cdef” where ‘a’ and ‘b’ are the Studio major and minor digits and ‘c’, ‘d’, ‘e’ and ‘f’ are the component’s major, minor, maintenance and patch digits. For example the patch release “3.5.0.1” for Studio 10.22 will be returned as “102.3501” by \$version. The same component when built for Studio 8.1.7 will return “81.3501”.

**Note:** Version number strings displayed by the system may have a slightly different format. These strings will typically display the component’s version number but further info may display the Studio version number in parenthesis.

## Checking the version programatically

It is possible to programmatically check component version numbers using the Omnis notation **\$components.component\_name.\$version**. We strongly recommend that Omnis Studio applications always check the version numbers of Brainy Data external components against the minimum that is required for the application to operate correctly. The best time to check the version is during startup, perhaps in the startup task. Most of our examples provide sample code for this.

Example:

```
; this example tests for OWrite version 5.6.0.0 or better and
; Studio 11.0
Calculate required_version as "110.5600"
; check Studio compatibility by dropping the fraction
If ( int(required_version) <> int($components.OWrite.$version) )
    OK message {OWrite is incompatible with this version of Studio}
End If
; check xcomp compatibility by comparing the fraction
If ( mod(required_version,1) > mod($components.OWrite.$version,1) )
    OK message {OWrite version 5.6.0.0 or better is required}
End If
```

This example code is very simplistic and it requires the developer to update this code when switching between different versions of Omnis Studio. The section “Supported Studio versions” demonstrates a more sophisticated approach by checking the studio version utilising the **sys(1)** function.

## Checking the file version of the DLL

Brainy Data external component DLLs include system resources on Macintosh and Windows that allows one to inspect the version number of the DLL in the file system. Unfortunately, due to technical limitations, these version numbers will only display the three digit component version number, but further system version info strings (typically following the copyright) may now display the Studio version number for which the component was build inside parenthesis.

To view the DLL version number on Macintosh, right-click the external component bundle and select “Get Info” from the context menu. The info window will display the version number as part of the Brainy Data copyright message.

On Windows, right-click the external component DLL and select “Properties”. On the properties window select the “Version” tab (“Details” tab in Windows 7). Both the “Copyright” and “Product Version” will display the component’s version number.

You may notice a capital letter ‘R’ or ‘D’ following the version number. The ‘D’ stands for time limited demo build and the ‘R’ stands for full release build which you will only receive when you have licensed our software.

## Supported Studio versions

We do not build different DLLs for every version of Studio. Below is the list of separate builds that we provide and the versions of Studio that each build covers. This list will be updated as and when Tiger Logic release new software that requires new builds.

Component Version	Studio Versions Supported
81.xxx(x)	All Studio 8.1 versions prior to version 10.0
100.xxx(x)	All Studio 10.0 versions prior to version 10.1
101.xxxx	All Studio 10.1 versions
102.xxxx	All Studio 10.2 and 10.22 versions
110.xxxx	All Studio 11.0 versions (until further notice)

When downloading demo software from our demo download page or release software from our support pages, the downloaded folders containing the DLLs are arranged according to the Studio version. For example the fat-client component of OWrite version 5.6.0.0 for Studio 10.1 versions prior to version 10.2 on Macintosh is located in the folder

```
owrite_5600_xxx_mac/studio_1010/xcomp
```

Studio version 10.2 components would be located in

```
owrite_5600_xxx_mac/studio_1020/xcomp
```

Armed with the above information, we can now programmatically check the component's version numbers while catering for all currently known versions of Studio. When new versions of Studio are used they will pass the test as long as the major versions match.

Example:

```

; get the studio major and minor version digits using sys(1)
; minor version is extracted as single digit so 8.1.7 will
; become 81 and 10.22 will become 102 (tmp is of type character)
Calculate tmp as sys(1)
Calculate studio_version as
    con(strtok(nam(tmp), '.'), mid(strtok(nam(tmp), '.'), 1, 1))
; get the studio version from the component you wish to test
Calculate tmp as $components.OWrite.$version
If (studio_version = strtok(nam(tmp), '.'))
Else
    ; put up message xcomp is not compatible with this version of studio
End if

; you may also now test the xcomp minimum version
; tmp now contains the xcomp version following the prior strtok call
If (5600 > int(tmp))
    ; put up message minimum version is oWrite 5.6.0.0
End if

```

## Updated External Components

The following is the current list of external components that employ the new version number format. The “Version” column displays the version number in which the new format was introduced. This table will be updated as we release new versions of our software that implement the new version format.

IMPORTANT NOTE: Since version 4.1.2.3, OWrite now has four digit version numbers following the decimal separator.

Component	Version	Example Versions (Studio Interface)
OWrite JS-OWrite	5.6.0.0	81.5600 - 100.5600 - 101.5600 - 102.5600 - 110.5600
PDFDevice	5.2.0.0	81.5200 - 100.5200 - 101.5200 - 102.5200 - 110.5200
OSpell2	3.5.1.0	81.3510 - 100.3510 - 101.3510 - 102.3510 - 110.3510
OGantt	4.4.0.0	81.4400 - 100.4400 - 101.4400 - 102.4400 - 110.4400
OCal	2.3.0.0	81.2300 - 100.2300 - 101.2300 - 102.2300 - 110.2300
JS-Signature	1.3.0.0	81.1300 - 100.1300 - 101.1300 - 102.1300 - 110.1300

## **Document History**

22 January 2024: Updated for Studio 11.0

20 October 2022: Updated for Studio 10.2

09 April 2020: Updated for Studio 10.1

02 July 2019: Updated for Studio 8.1 and 10.0

08 June 2016: Updated for Studio 8.0

02 September 2015: Updated for Studio 6.1

05 September 2013: Updated for OGantt version 4

14 August 2013: Updated for PDFDevice version 3

29 April 2013: minor grammatical corrections and signpost to OCal release notes

25 April 2013: first publication