Brainy Data

Newsletter

You are receiving this Newsletter because you expressed an interest in our software or are currently licensing one or more of our software components. This newsletter provides up-to-date information on the progress of our current developments, new software releases, and details about future developments.

Please also visit our news page for a summary of developments over the past twelve months.

https://home.brainydata.com/news.htm

NOTE: NEW SUBSCRIBERS will automatically receive a copy of the last newsletter that we circulated and consequently the date shown at the top of the newsletter may not be current.

Introduction

Welcome to our final Newsletter ...

... for 2022. This year has been especially busy. Besides a number of intricate maintenance releases, we have been involved with various new ports which we hope will be completed in 2023. More of that below. This Christmas, our offices will be closed until the 18th of January 2022. We at Brainy Data wish everyone a heartfelt

Merry Christmas and a Happy New Year!

PDFDevice version 5 Release

The new static $file... functions

Version 5 implements new static functions that accomplish the merging, signing and encryption of existing PDF files[1]. These new functions are listed in the Omnis Catalog. There is also a new report object for creating PDF files with AcroForm fields and a new static function to read data from form fields in PDF files.

Release Notes: https://supportpublic.brainydata.com/rn/pdfdevice_5000.pdf

Feature Overview: https://products.brainydata.com/pdfdevice/v5_features.htm

Latest Examples and Documentation

There are additional v5 and v4 PDFDevice examples and new updated documentation all of which can be downloaded or viewed via the links below

V5 Documentation: https://supportpublic.brainydata.com/documentation.htm

V5 Examples: https://demos.brainydata.com/software/pdfdevice_and_podofo.zip

V4 Examples: https://demos.brainydata.com/software/pdfdevice_and_jsclient.zip

Final Release supports Studio 8.1.7 or better

PDFDevice version 5 was an especially difficult feature release involving some complex open source software. For a while we were unable to build for Studio versions prior to Studio 10.2. However, after some further efforts, the final release now supports Studio 8.1.7 or better on both Macintosh and Windows. Unfortunately, PDFDevice version 5 cannot support Studio versions prior to version 8.1.7. We will continue to maintain PDFDevice version 4 for backward compatibility, but new features that rely on the open source will be restricted to version 5 and thus Studio 8.1.7 or better. Please also read about Studio 11 ports in the section “Current projects” below.

Linux Support

We had hoped to include Linux as a new port, but have not been able to complete this port this year due to a high demand on support and maintenance in the later part of the year and we were forced to release version 5 without linux support. Read more about this port in the section “Current projects”.

1. Please be aware we can only guarantee that these features will work fully with PDF files produced by PDFDevice, although tests have not shown any issues with PDF files that were produced by other tools.

Other Feature & Maintenance Release

This section lists all minor feature and maintenance releases. **Please make sure you read the release notes and perform your own tests prior to distributing our new software.**

oWrite/jsoWrite version 5.4.0.0

Minor feature release: Features new grammar options as well as important changes to the properties $papercontinuous and $newprimeasure and although localised were substantial and require thorough testing.

Release Notes: https://supportpublic.brainydata.com/rn/owrite_jsowrite_5400.pdf

Download Locations: -

- Desktop: https://support.brainydata.com/owrite_su.htm
- JS Server/Client: https://support.brainydata.com/jsowrite_su.htm

oWrite/jsoWrite version 5.3.0.0 & 5.3.0.1

Minor maintenance release: Changes centred mainly around $papercontinuous which were applied across two releases. Even if you intend to skip these in favour of 5.4.0.0, make sure you read the release notes for this release carefully.

Release Notes: https://supportpublic.brainydata.com/rn/owrite_jsowrite_5300.pdf
Download as for 5.4.0.0.

oWrite/jsoWrite version 5.2.0.0

Minor feature release: Implements web links for picture objects as well as improvements to return RTF for table cell calculations. Software corrections were centred around page number and page count fields.

Release Notes: https://supportpublic.brainydata.com/rn/owrite_jsowrite_5200.pdf
Download as for 5.4.0.0.

pdfDevice 4.1.2.0

Minor maintenance release: Resolves a crash on new M1 and M2 machines and fixes a strange situation involving the strip duplicate image option.

Release Notes: https://supportpublic.brainydata.com/rn/pdfdevice_4120.pdf
Download: https://support.brainydata.com/pdfdevice_su.htm

oGantt 4.3.1.0

Minor maintenance release: Resolves a problem with $gmovephase and suspensions.

Release Notes: https://supportpublic.brainydata.com/rn/ogantt_4310.pdf
Download: https://support.brainydata.com/ogantt_su.htm

jsoCal 1.1.0.0

Minor feature release: Fixes numerous minor but important issues and we strongly recommend that you upgrade to this version. There are also new navigation buttons and the appearance of the year view has been much improved.

Release Notes: https://supportpublic.brainydata.com/rn/jsoCal_1100.pdf
Download: https://support.brainydata.com/jsocal_su.htm

Current Projects

jsoGantt

In 2023 we will renew our effort to complete the jsoGantt port which had been discontinued on several occasions in the past. The project has now been planned out in some detail and we plan to begin in January 2023.

Linux Ports

As already mentioned above, we had hoped to release a Linux version of pdfDevice version 5 before Christmas. Unfortunately, a number of difficult support cases meant that we could not complete this port. We will try again in the new year. Although the jsoGantt port has been scheduled for early 2023, we hope we have sufficient capacity to work on this port alongside the jsoGantt port.

The PDFDevice port, will be followed by other ports (jsoWrite and jsoGantt) soon thereafter. Of course, being entirely developed using the JSON control interface, jsoCal can already be deployed on linux servers.

OWrite/jsoWrite version 6

There are a number of new features in the pipeline for oWrite version 6 which will include

- Export documents direct to PDF
- AcroForm fields within oWrite documents
- An enter data mode for oWrite documents so users can safely enter data directly into documents via form fields

Dates are to be announced.

Studio 11 ports

Studio 11 ports for the Studio beta release will be carried out on request (requires a full Developer Support Subscription). This port will be a little more challenging than previous ports as we will need to upgrade our Visual Studio and XCode build environments. This may also mean that we may no longer be able to support some of the older Studio versions, although initial investigations indicate we can at least continue to support Studio 8.1.7 onwards.

Documentation & Technical Notes

Documentation

We have updated the documentation for oWrite, jsoWrite, pdfDevice, and oCal. Some of this work was substantial, especially for the new oCal and jsoCal documentation which has now been combined into one document.

The oCal documentation had major new sections added, the pdfDevice documentation has been updated to document the new version 5 features and this year's oWrite and jsoWrite improvements have now been fully documented.

You can view or download documentation at https://supportpublic.brainydata.com/documentation.htm.

Technical Notes

TN0034 is a new technical note and lists the most common error codes that may be returned by the printer driver installation or during printing to PDF.

TN0022 has been updated for Studio 10.22.

Technical Hints

The 'Technical Hint' section documents one or more interesting software intricacies that may have come up in technical support queries or that we encountered during research and development.

We have also produced the first of an annual ‘Technical Hint’ Compendium which can be downloaded from the links below. This compendium collects all technical hints that we ever published in a handy single PDF file.

HTML: https://home.brainydata.com/news/compendium/html/compendium.html
PDF: https://home.brainydata.com/news/compendium/pdf/compendium.pdf

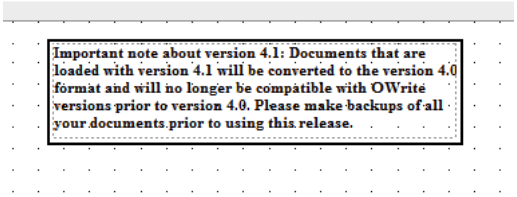
oWrite: $newprimeasure

It seems that this feature has largely been ignored by developers (we know this because it wasn’t actually working for some time), so we thought we would point out its benefits, now that it is working properly. This article assumes that you will be using oWrite version 5.4.0.0 which includes an important fix and some further improvements. The purpose of this property is to overcome the device dependency of Omnis Studio screen reports.

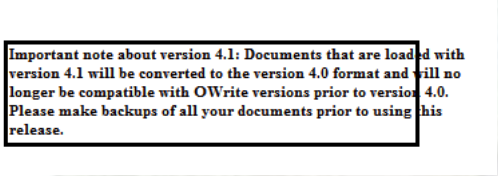
When oWrite’s $printdpi is set to zero (recommended) or perhaps 600, to closely match some printer devices, oWrite will measure document text at a resolution that is more compatible with high resolution printers and draws text one character at a time on screen, advancing the print position of each character according to the higher resolution. This results in oWrite documents being rendered in what is called the WYSIWYG (what you see is what you get) display mode. In other words how the document is displayed on screen (regardless of screen resolution) is how it will appear on paper when printed or how it is displayed on other high resolution devices by competent renderers, such as Adobe PDF Reader. The problem is that Studio uses the device dependent method of calculating font sizes for rendering to screen and as the rendering is integer based, there is a loss of accuracy resulting in differences of text appearance between device resolutions. When oWrite prints documents to screen, it can result in words either running into each other or large gaps appearing between words as oWrite word measurements would differ to Studio measurements. OWrite has to add a continuous stream of text to the print manager as it would be a huge drain on performance and system resources to add a character at a time. Back in Studio 5.2, we arranged with Omnis Software (back then Raining Data), to add a feature to the Studio print manager that allows the positioning of text on a character basis with measurements being supplied at a resolution of 1000 DPI. Text is still added as a continuous stream of text, but this stream now also is sent with an array of additional measurements that Studio uses for rendering words one character at a time, simulating what oWrite does when it displays documents on screen.

To appreciate the problem on MS Windows, try adding some Times New Roman 8 point bold text to a report, draw a box around it, then print it both to screen and printer. You will see a substantial difference. See images below...

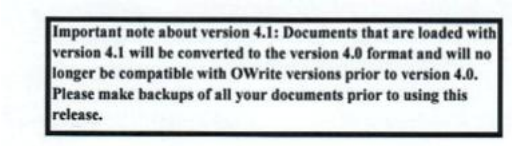
Studio report



Studio screen report



Scan of printed output



You can see how the screen report renders the text too large and how the report class wraps the text differently.

Rendering the same text with oWrite produces identical output within oWrite documents, screen reports and on paper, as long as you turn on $newprimeasure.

Why does Studio get it wrong? Studio screen reports on windows are traditionally rendered at 96 DPI. When specifying font sizes they are specified as points (72 DPI). To render 8 point text to screen on windows, Studio has to multiply 8 by 96 and divide it by 72, which would result in a font size of 10.6666 for 96DPI screen devices, but must choose either a 10 or 11 point font to measure and render text, resulting in the text being drawn slightly larger or smaller depending on whether one rounds up or down. The longer a line of text is, the greater the inaccuracy that will accumulate.

Why does oWrite get it right? OWrite simply measures text at the font's units per eM (UPM) when $printdpi is zero (the recommended option), which can be as large as 1000 or 2048 UPM, and scales down to the specified point size and screen DPI. To measure an 8 point font, oWrite would simply measure at the font's UPM and scale down the measurement for each character using floating point variables. The measurement is scaled according to the intended point size and device resolution and the position for each character calculated using floating point accuracy. This at worst results in the odd character exhibiting a partial pixel inaccuracy, where two adjacent characters may slightly crowd or repel each other. On low resolution devices this may be just about noticeable, but negligible producing reasonable WYSIWYG output. On higher resolution screens (i.e. retina screens), this will provide near perfect WYSIWYG output.

Why did we choose 1000 points as the basis for the $newprimeasure? The answer is that it is the standard for providing character widths in PDF and PDF is the most common cross-platform digital document format.

The $curobjdisplay property for kWriObjTypeInfo fields

The current documentation only talks about $curobjdisplay for calculated fields, but this property is also used with info date and time fields. For these info fields it can provide alternative formatting strings for displaying the date and time. Currently, oWrite only supports MS Word style date and time formatting strings using a combination of 'd', 'M', and 'y' characters for date fields and a combination of 'H', 'h', 'm', 's', 'A' and 'a' characters for time fields. For example the string "dddd, dd MMM yyyy" displays "Thursday, 15 Dec 2022" and the string "HH:mm.s" displays the time "16:45.20". The following is a detailed description of MS Word date and time formatting codes:

Date Codes

d	single or two digit date, i.e. '7'
dd	two digit date, i.e. '07'
ddd	abbreviated name of day, i.e. 'Tue'
dddd	full name of day, i.e. 'Tuesday'
M	single or two digit month, i.e. '1'
MM	two digit month, i.e. '01'
MMM	abbreviated name of month, i.e. 'Jan'
MMMM	full name of month, i.e. 'January'
yy	two digit year
yyyy	four digit year

Time Codes

H	single or two digit hour, i.e. '5'
HH	two digit hour, i.e. '05'
m	single digit minute, i.e. '2'
mm	two digit minute, i.e. '02'
s	two digit seconds
am/pm	lower case am/pm suffix, i.e. 'am'
AM/PM	upper case am/pm suffix, i.e. 'AM'

⚠ IMPORTANT: New Bank Account Details

Our international payment IBAN and SWIFT BIC codes have changed. For the time being, the previous codes will continue to work, but we urge you to update your records prior to making your next payment.

The new bank details are available here

https://shop.brainydata.com/howtopay.pdf

You will require the password for the PDF which you can find in a previous email that accompanied our invoice. If you cannot find it, please contact our admin department.

Important Links

News: https://home.brainydata.com/news.htm
Products & pricing: https://products.brainydata.com
Demo/Examples Downloads: https://demos.brainydata.com/download.htm
GitHub: https://github.com/BrainyData
Sponsors: https://home.brainydata.com/sponsors.htm
Feedback: https://home.brainydata.com/customers.htm

Online Documentation: https://supportpublic.brainydata.com/documentation.htm
Technical notes: https://supportpublic.brainydata.com/technotes.htm
Support Request Form: visit https://supportpublic.brainydata.com and click “Software Downloads”
Software Downloads: visit https://supportpublic.brainydata.com and click “Contact Support”