PRINTER PLUGIN

User manual – v1.2.1 © Baraonda 2022-2023

How it works

The plugin uses a step-by-step approach to send instructions to the printer. To print a text, you first set the position on paper (think of it as a virtual cursor), then you can set optional text attributes and finally the text is sent to the printer. All coordinates are in millimeters, this allows to obtain the same result on different printers regardless of the DPI of the device.

Quick start

After installing the package, you need to add a "using PrintLib;" at the top of your script and you have to create a "printer = new Printer();" somewhere in your code. Just remember to keep a reference to the printer object, because you may need that later. The smallest piece of code to start would be something like this:

```
using PrintLib;

public class PrintTest : MonoBehaviour
{
    Printer printer;

    void Start()
    {
        printer = new Printer();
        printer.StartDocument();
        printer.SetPrintPosition(50, 50);
        printer.PrintText("Hello World!");
        printer.EndDocument();
    }
}
```

As you may guess, the two functions StartDocument() and EndDocument() must always surround your printing code, as they allow to initialize the printer and to send all the data to the printer to actually print the text.

Coordinates system

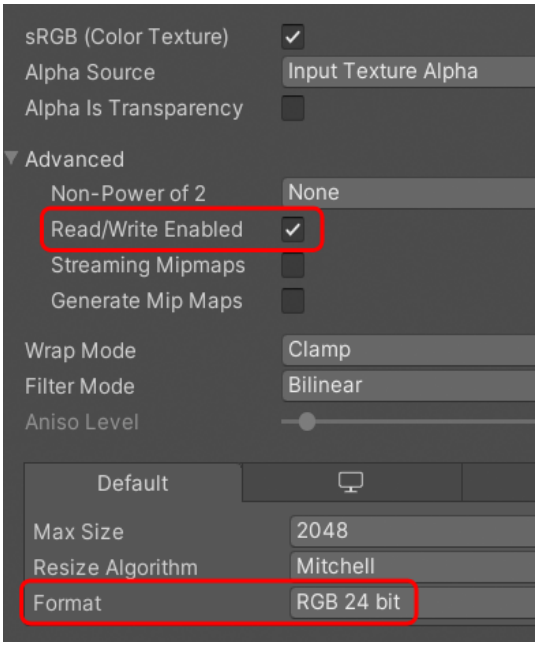
The unit measure for SetPrintPosition and for image and font sizes are expressed in millimeters, with the origin at the **top-left** corner of the page.

So, in the previous example, the “Hello World!” would be printed at 50mm (5cm) from left border and 50mm from top border.

Note: use the GetPageSize() method to obtain the total size, in millemeters, of the page you are printing on.

Printing images

To print an image just use the PrintTexture() function. Remember that textures must have the Read/Write flag enabled and must be uncompressed (RGB 24 bit or RGBA 32 bit formats).



Alternatively, you can print an image directly from disk, using the PrintImageFromFile() function and passing a string with the path. The file doesn’t need to be in the assets or resources folder, but can be anywhere on disk.

```
printer.PrintImageFromFile("C:/MyFile.png", 30, 30);
```

When printing images, only one size can be specified as parameter. This is useful if you want to scale an image, but you don’t want to calculate the proportions by yourself. Just pass one size in this case and the other will be calculated proportionally.

printer.PrintTexture(texture, 80);

Printing text

Text printing offers more options, in the following example we print a string using almost all the available options.

```
printer.SetPrintPosition(50, 50);
printer.SetTextFontFamily("Verdana");
printer.SetTextFontSize(6);
printer.SetTextColor(Color.blue);
printer.SetTextFontStyle(TextFontStyle.BoldItalic);
printer.PrintText("Hello World, but better!");
```

Use **PrintText("text")** if you want to print single text lines. This is useful when you know how long the string is, and that the lines won't exceed the page margin.

If you have to print more characters you may pass two additional parameters to the PrintText() function: width and height. In this way you will define an area inside which the text will be wrapped in. In the latter case you can also align your text with TextHorizAlignment (Left, Center or Right) and TextVertAlignment (Top, Center, Bottom) for vertical alignment as well.

This is an example (writes the text inside a 150x150mm rect):

```
printer.PrintText(text, 150, 150, TextHorizAlignment.Center, TextVertAlignment.Center);
```

About Fonts

Remember that since font size is in millimiters, a size of 6 does not correspond to a traditional 6 points font but to a 6mm height font (that is instead quite big).

When choosing a font, you DON'T need to install that font in your Unity project, you just need that font installed on the target system (among the installed Windows fonts).

Printer settings

After the SelectPrinter method and before StartDocument, you can change many printer settings with this method:

```
printer.SetPrinterSettings(Orientation.Portrait, PaperFormat.A4, 300, ColorMode.Color, 1);
```

For any of the parameters you can pass the Default value (or zero in some cases) if you don't want to change that particular setting.

The settings you can change, before printing, are:

- Page orientation (portrait or landscape)
- Page format (A4, A3, A5, Letter)
- DPI
- Color mode (color or monochrome)
- Number of copies

Alternatively, you can call the **printer.ShowPrinterDialog()** method which shows a configuration form, allowing the user to personally edit the printer settings.

The **printer.ResetPrinterSettings()** can be called anytime to reset the original printer configuration.

Reference

`int GetPrinterCount()`	Get the number of printers installed on system.
`string GetPrinterName(int index)`	Get the name of a printer at a given index, the total number of printers is given by `GetPrinterCount()`.
`string GetDefaultPrinterName()`	Get the default printer.
`void SelectPrinter(string name)`	Select a printer using a string obtained by `GetPrinterName()`. If you don't select a specific printer, the default printer is used. Note: select the printer **before** running `StartDocument()`.

`int SetPrinterSettings(`
`  Orientation orientation,`
`  PaperFormat paperFormat,`
`  int DPI,`
`  ColorMode colorMode,`
`  int copies)`
	Call this function before StartDocument.
	- **Orientation**: portrait or landscape
	- **PaperFormat**: A4, A5, A3…
	- **DPI**, set an int value like 300 or 400, zero for default
	- **ColorMode**: Color or Monochrome
	- **Copies** (1 or more) or leave zero for default

`int ShowPrinterDialog()`	Show a printer settings dialog, so that the user can change all available printer settings as needed before printing. Returns -1 (NOT_OK) even if the user hits the 'Cancel' button. Call this method after `SelectPrinter()` and before `StartDocument()`.
`void ResetPrinterSettings()`	Restore the original printer configuration.

`void GetPageSize(`
`  ref int width_mm,`
`  ref int height_mm)`
	Get the current page size in millimeters.

`string GetLastGdiStatus()`	Get the last system error, if something went wrong.
`void StartDocument()`	Initializes the printer and start receiving commands to print elements.

`void SetPrintPosition(`
`  float left_mm,`
`  float top_mm)`
	Move the virtual printer cursor to a given position, in millimeters, from the top-left corner of the page.

`void NewPage()`	Create a new page (or add a page break).
`void EndDocument()`	Finish the current document, and send all data to the printer.

`int PrintTexture(`
`  Texture2D texture,`
`  float width_mm,`
`  float height_mm)`
	Print a Texture2D object. The texture must have Read/Write flag enabled and must be uncompressed.
	If one of the two size params is zero, it will be auto-calculated using the provided one.

`int PrintImageFromFile(`
`  string path,`
`  float width_mm,`
`  float height_mm)`
	Print an image from a path.

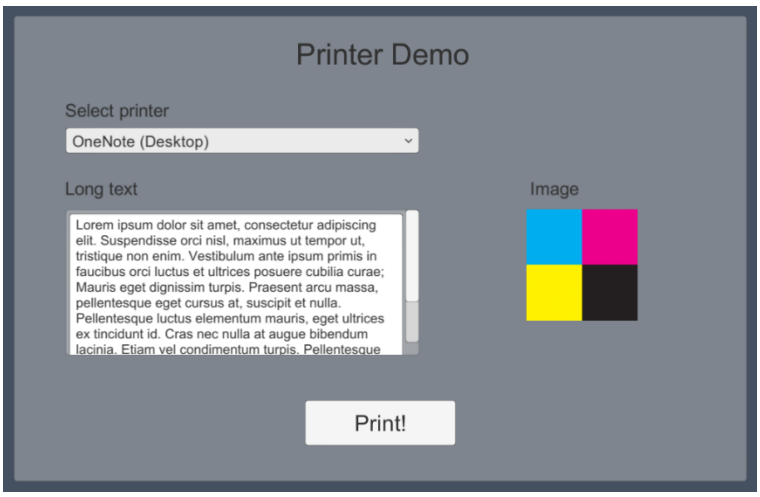
`int PrintText(`
`  string text,`
`  float width_mm,`
`  float height_mm,`
`  TextHorizAlignment alignment,`
`  TextVertAlignment vAlignment)`
	Print a text. If you specify width_mm and height_mm the text will be wrapped inside a rect.
	You can optionally specify text alignment for multi-line texts, when the text rect is also specified.

`void SetTextColor(` `  Color color)`	Set text color.
`void SetTextFontSize(` `  float size_mm)`	Set font size (height), in millimeters.
`void SetTextFontFamily(` `  string name)`	Set font familiy.
`void SetTextFontStyle(` `  TextFontStyle style)`	Set font style.

Samples

Please, open the scene located at BrnPrinterPlugin/Scenes/PrinterDemo and the script Scripts/PrintDemo.cs to see more printing examples.

In the Demo scene you will also see a simple way to enumerate and select a printer.



Troubleshooting

If you get “dll not found” error in Unity try to install the Microsoft Visual C++ 2015 Redistributable package from this link:

https://www.microsoft.com/en-US/download/details.aspx?id=52685

Here’s a version with more details:

https://learn.microsoft.com/en-US/cpp/windows/latest-supported-vc-redist?view=msvc-170

You may need to install this package on your final user’s PC, to run the build correctly. You can install the package silently by using /Q parameter, in this way:

“.\vc_redist.x64.exe /Q”

Contacts

Don’t hesitate to write me for help and questions at support@baraonda.eu