IDPhotoCapture

IDPhoto Remove Background

User Guide

2024

Introduction

IDPhoto Remove Background is a console application designed to replace background on ID photos. Below is an example command line to process a file.

Copy and use it to process your photo (specify your paths for the -i and -o keys).

To copy, select the code and press **Ctrl +C**.

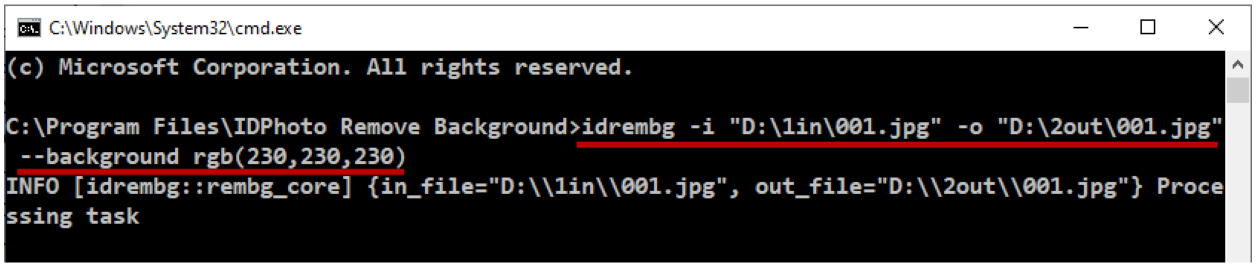
Example:

```
>idrembg -i "D:\1in\001.jpg" -o "D:\2out\001.jpg" --background rgb(188,188,216)
```

-i – path to the source image

-o – path to the folder where you want to save the processed image

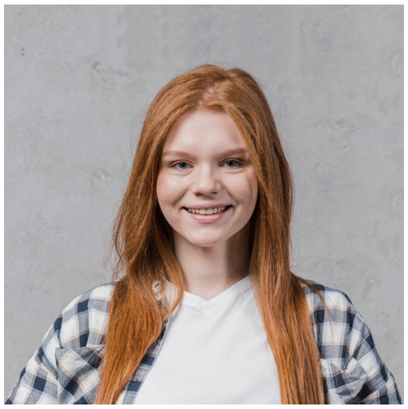
--background – background color



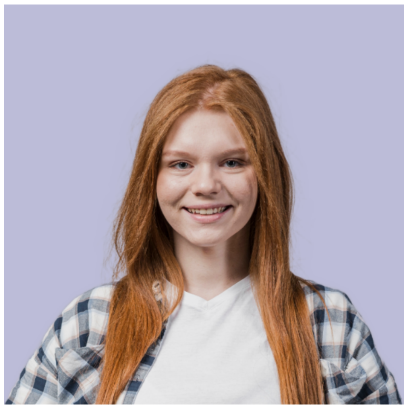
```
(c) Microsoft Corporation. All rights reserved.

C:\Program Files\IDPhoto Remove Background>idrembg -i "D:\1in\001.jpg" -o "D:\2out\001.jpg"
 --background rgb(230,230,230)
INFO [idrembg::rembg_core] {in_file="D:\\1in\\001.jpg", out_file="D:\\2out\\001.jpg"} Proce
ssing task
```

The command will save the processed image to the folder specified in the -o key.

Source photo:



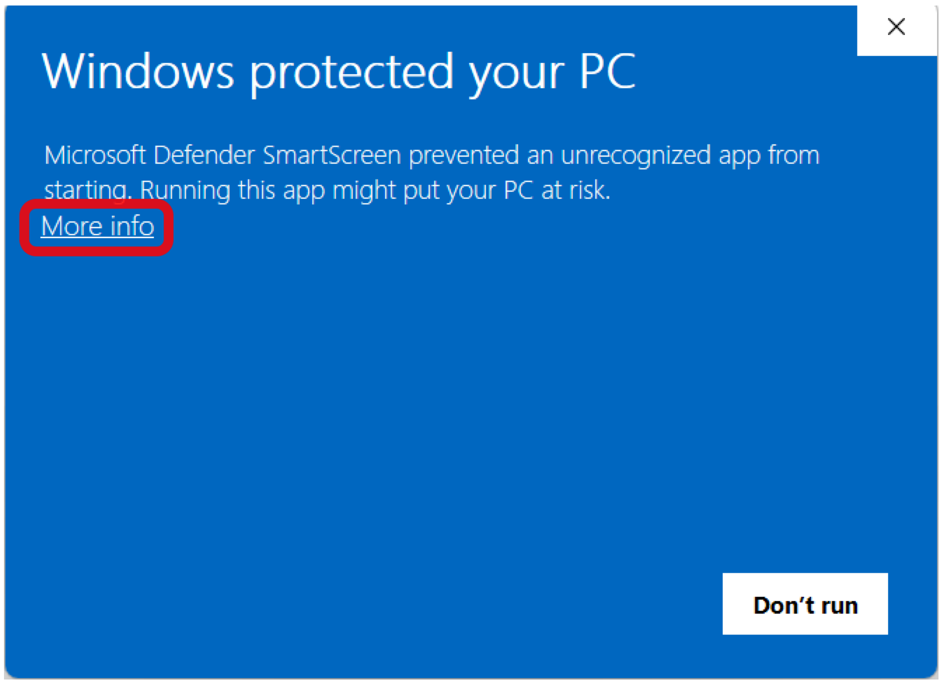
Processed photo:



Further in this document you will find instructions and examples to quickly get started with **IDPhoto Remove Background**.

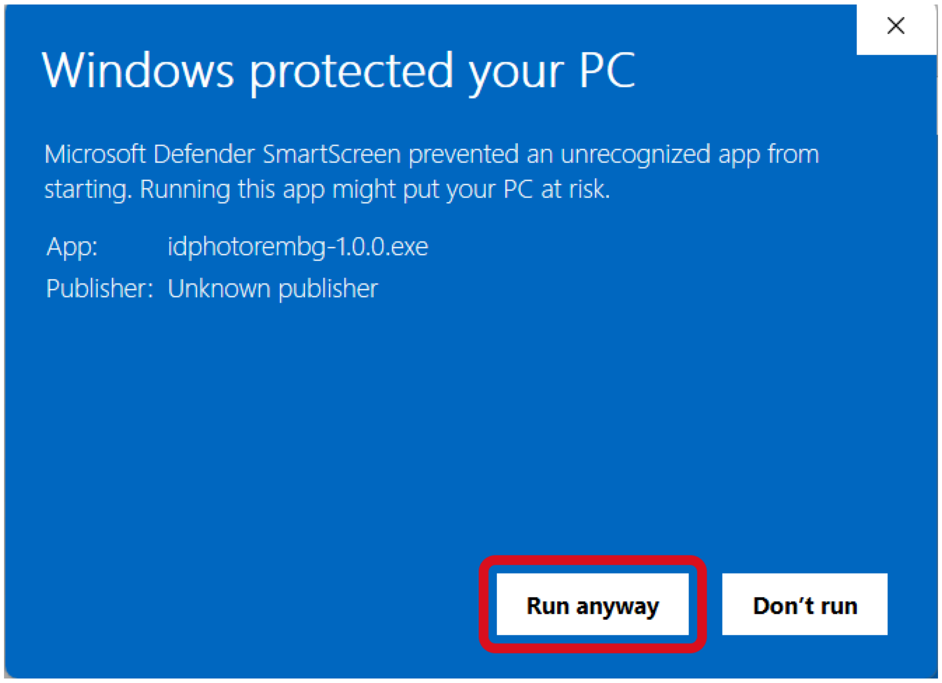
Installation

1. Download the installer from our website: idphotocapture.com/downloads
2. Run the installer
3. In the Windows security notification, allow the application to run. To do this, click **More info** -> **Run away**.



Windows protected your PC

Microsoft Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.


Windows protected your PC

Microsoft Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.

App: idphotorembg-1.0.0.exe

Publisher: Unknown publisher

4. Then follow the instructions of the installer.
5. By default, the application will be installed in the folder C:\Program Files\IDPhoto Remove Background
6. After installation is complete, IDPhoto Remove Background folder will contain the following files:
   - samples – folder with sample images
   - idrembg.exe – **IDPhoto Remove Background** application
   - integration.exe – utility that ensures compatibility of **IDPhoto Remove Background** with inPhoto products
   - license_idphotocapture.txt – licence agreement
   - licensing.exe – activation utility
   - uninst.exe – **IDPhoto Remove Background** uninstaller

Activation

Upon installation, the application operates in **Demo mode**. In this mode, all processed images will display a **Demo** watermark. To remove this watermark and unlock the application, you must activate your product license.

To activate the application, follow the steps below:

1. Run `C:\Program Files\IDPhoto Remove Background\licensing.exe`

2. In the activation utility window that opens, click on Activate

3. In the next step, enter your license serial number (license key) and click Activate. To obtain your serial number, purchase a license for the application. After completing your purchase, the serial number will be provided to you via email or displayed on the purchase confirmation page. The serial number is formatted as five blocks of four digits, separated by hyphens. For example: **9999-9999-9999-9999-9999**.

4. The activation of your serial number on the server will now begin. This process typically takes a few seconds, but may take up to a few minutes. Please wait for the activation to finish.

5. Once activation is successfully completed, the utility will display a confirmation message. You can now use the **IDPhoto Remove Background** feature without any restrictions. Additionally, you are free to create copies and relocate the `idrembg.exe` file to other folders while maintaining the activation.

If there is no Internet connection on your computer, you can activate the application **offline**.

Keys

-h or --help

Get information about application keys by running the idrembg command with the -h or --help key.

Example:

```
>idrembg -h
```

or

```
>idrembg --help
```

Running such a command line will cause the application to display key information on the console.

The table below provides a description of each key along with an example input.

.\1in – folder containing source files
.\2out – folder for saving processed files

-h or --help – displays information about available keys

Example:

```
>idrembg -h
```

or

```
>idrembg --help
```

-i or --in-file <path to source file> – processes a single file (image) and saves it in the folder where the source file is located if the -o key is not specified.

Example:

```
>idrembg -i D:\1in\001.jpg"
```

or

```
>idrembg --in-file "D:\1in\001.jpg"
```

– processes the 001.jpg file and saves it to the D:\1in folder named 001.idrembg.jpg. The default background color is light grey.

-o or —out-file <path to processed file> – specifies the location, name and format of the processed file

Example:

```
>idrembg -i "D:\1in\001.jpg" -o "D:\2out\001.png"
```

or

```
>idrembg --in-file "D:\1in\001.jpg" --out-file "D:\2out\001.png"
```

– processes the file 001.jpg and saves it to the folder D:\2out in **PNG** format with the name 001.png. The background color is set to light grey by default.

--in-dir <path to folder> – processes in the specified folder all graphic files with extensions: **jpg**, **jpeg**, **png**, **webp**, **bmp**, **tif**, **tiff**. If the --out-dir key is absent, it saves the processed files to the folder where the original files are located.

Example:

```
>idrembg --in-dir "D:\1in"
```

– processes graphic files in the D:\1in folder and saves them to the same folder
For example, the D:\1in folder contains files 001.jpg, 002.jpg and 003.jpg. After the command is executed, the original files will be saved in the D:\1in folder and the files 001.idrembg.jpg, 002.idrembg.jpg, 003.idrembg.jpg will be created.
The background color is set to light grey by default.

--out-dir <path to folder> – saves the processed files to the specified folder

Example:

```
>idrembg --in-dir "D:\1in" --out-dir "D:\2out"
```

– processes all graphic files with extensions: **jpg**, **jpeg**, **png**, **webp**, **bmp**, **tif**, **tiff** in the D:\1in folder. The processed files are saved to the D:\2out folder. For example, the D:\1in folder contains files 001.jpg, 002.jpg and 003.jpg. When the command is executed, the original files will be saved in the D:\1in folder, and the files 001.jpg, 002.jpg, 003.jpg will be created in the D:\2out folder.

The background color is set to light grey by default.

-b or —background <color> – sets the background color

Example:

```
>idrembg -i "D:\1in\001.jpg" --background rgb(208,227,239)
```

– processes the 001.jpg file, saves it to the D:\1in folder named 001.idrembg.jpg and sets the processed image to a light blue background color (rgb(208,227,239)).

--overwrite – overwrites the processed file if it already exists. Files with the .idrembg suffix are always overwritten.

Example:

```
>idrembg -i "D:\1in\001.jpg" -o "D:\2out\001.jpg" --overwrite
```

– processes the file 001.jpg, saves it to the folder D:\2out named 001.jpg and overwrites it if the file D:\2out\001.jpg already exists.

--jpeg-quality <value> – sets the degree of compression (compression) of the processed image (only for JPEG files). The smaller the value in the key, the lower the quality of the image and the smaller its file size.

Example for the file:

```
>idrembg -i "D:\1in\001.jpg" --jpeg-quality 80
```

– processes file 001.jpg, saves it to the folder D:\1in with the name 001.idrembg.jpg. The processed file 001.idrembg.jpg will be created with a compression ratio of 80, i.e. its quality will be lower and its file size will be smaller than the original 001.jpg.

Example for the folder:

```
>idrembg --in-dir "D:\1in" --jpeg-quality 80
```

– processes all files in the folder D:\1in, saves them to the same folder with names containing the suffix .idrembg (for example: 001.idrembg.jpg). The processed files will be created with compression level 80, i.e. their quality will be worse and the file size will be smaller than the original files.

--output-format <format> – sets the format of processed files. Possible values: **jpg**, **jpeg**, **png**, **webp**, **bmp**, **tif**, **tiff**.

Example:

```
>idrembg --in-dir "D:\1in" --out-dir "D:\2out" --output-format png
```

– processes all files in the D:\1in folder and saves them to the D:\2out folder in **PNG** format

--show-system-info – displays system information

Example:

```
>idrembg --show-system-info
```

As a result of executing the command, the console will display system information about the application, including the serial number (if the application is activated).

To run the idrembg command, it is mandatory to set only the keys containing the path to the source file or folder: -i (--in-file) or --in-dir. You can also use these keys together, as shown in the example below. In this case, both the file and the folder will be processed.

Example:

Process file 001.jpg in folder D:\1in and all files in folder D:\3in.

```
>idrembg -i "D:\1in\001.jpg" --in-dir "D:\3in"
```

This command line will create a processed file D:\1in\001.jpg next to the source file D:\1in\001.idrembg.jpg, and will create processed files (e.g. 001.jpg, 002.jpg and 003.jpg) next to the source files (e.g. 001.idrembg.jpg, 002.idrembg.jpg and 003.idrembg.jpg) in the D:\3in folder.

Other keys can be configured according to your preferences.

Single Image Processing

Process an Image with One Single Key

To replace the background in an image, you just need to run the application with a one key -i (or --in-file) <path to source file>.

Example:

Let's start processing the 001.jpg file located in the D:\1in folder:

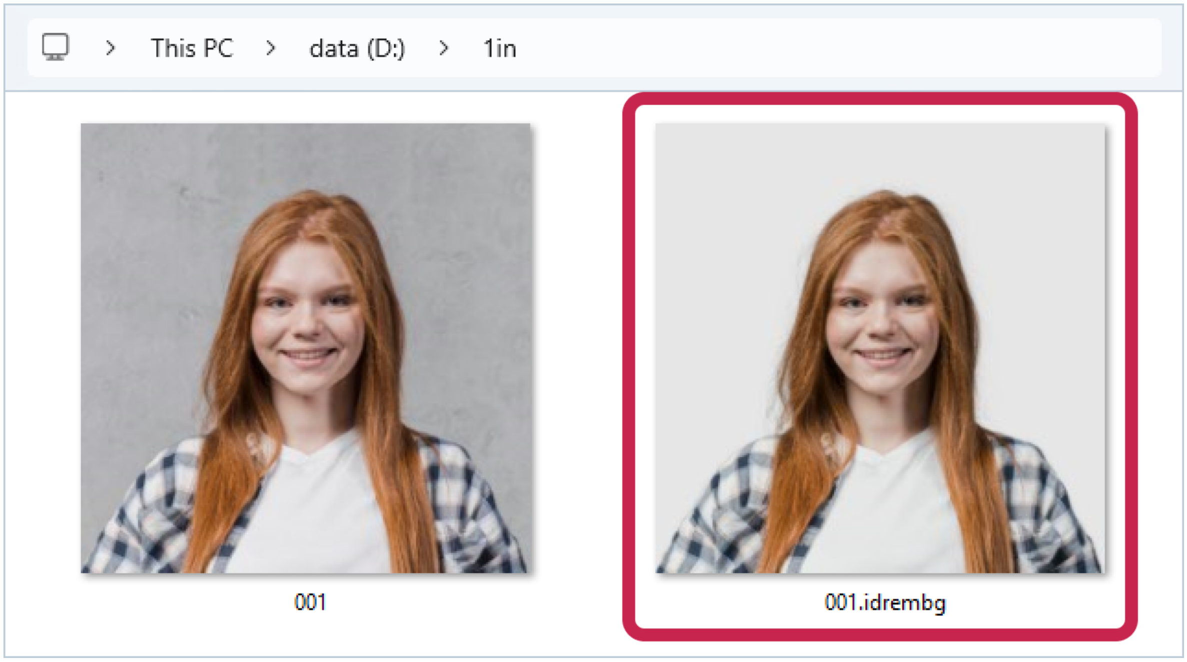
```
>idrembg -i "D:\1in\001.jpg"
```

or

```
>idrembg --in-file "D:\1in\001.jpg"
```

This command line will create a processed image next to the original image in the D:\1in folder. The name of the processed image will consist of the name of the original image and the suffix .idrembg - 001.idrembg.jpg.

The background color will be changed to light grey (rgb(230,230,230,230)). This color is always set by the application unless the key -b (--background) is specified.



Saving a Processed Image

The utility can be configured to save the processed image to a specified folder, with a particular name and format, by using the -o (or --out-file) key.

Example:

In the folder D:\1in there is a file 001.jpg, let's process it, and in the key -o set the folder where to save the processed file – D:\2out, its name and format – IDPhoto_001.png.

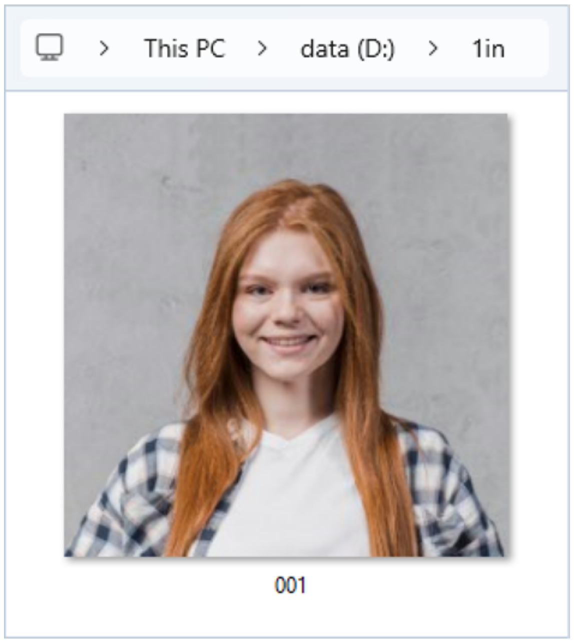
```
>idrembg -i "D:\1in\001.jpg" -o "D:\2out\IDPhoto_001.png"
```

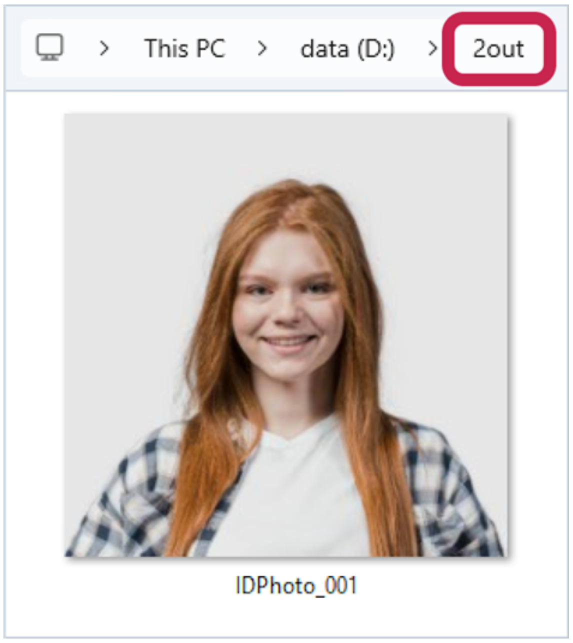
or

```
>idrembg --in-file "D:\1in\001.jpg" --out-file "D:\2out\IDPhoto_001.png"
```

As shown in the screenshot below, the processed file is saved with the name and format specified in the value of the -o (or --out-file) key.

As previously mentioned, if the background color is not specified, the application defaults to setting it to light grey.





Background Color

To set a background color on the processed image, use the -b or --background <color> option. The color can be specified in any **CSS**-supported format, including those with transparency (alpha channel), such as hex, rgb (rgba), hsl (hsla), and named colors.

A list of colors commonly used as backgrounds in ID photos is provided at the end of the manual 38.

Example:
Let's process file 001.jpg and specify that the background in the output image should be a light blue color:

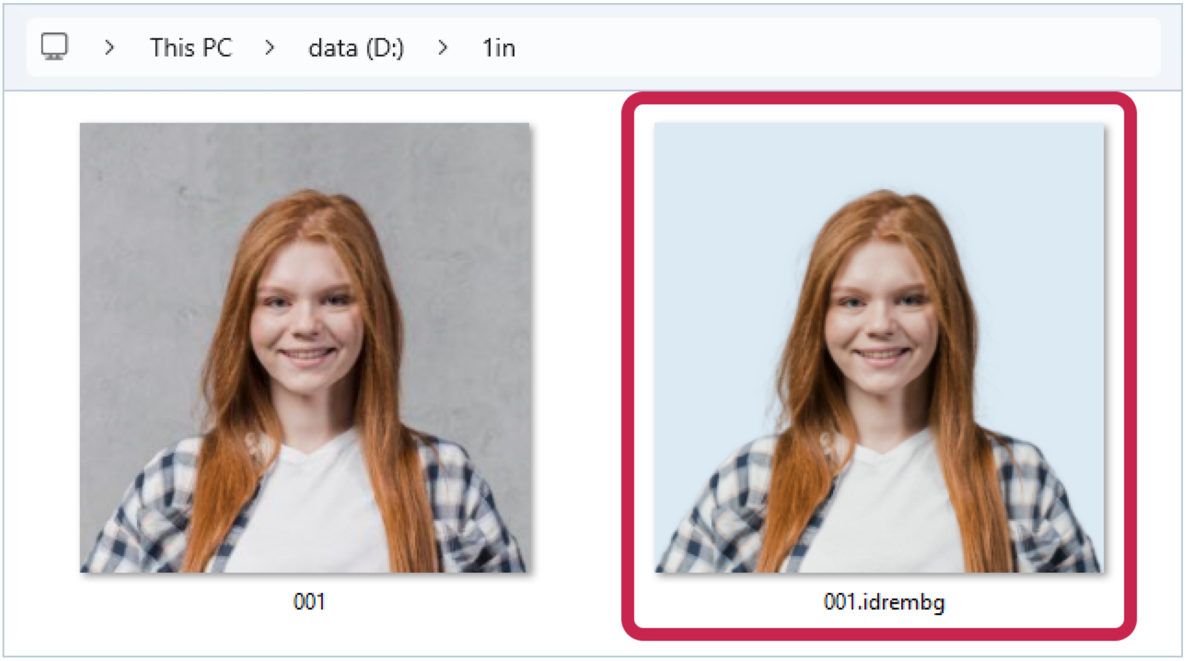
HEX

```
>idrembg -i "D:\1in\001.jpg" -b #DCEAF5
```

RGB

```
>idrembg -i "D:\1in\001.jpg" -b rgb(220,234,245)
```

This command line will create a processed image 001.idrembg.jpg with a light blue background in the D:\1in package.



Transparent Background

To create a processed image with a transparent background, the application must save it in a format that supports transparency, such as **PNG** or **TIFF**.

By default, the application preserves the format of the source file for the processed image. If original file is already in **PNG** or **TIFF** format, no further adjustments to file format are needed to maintain transparency.

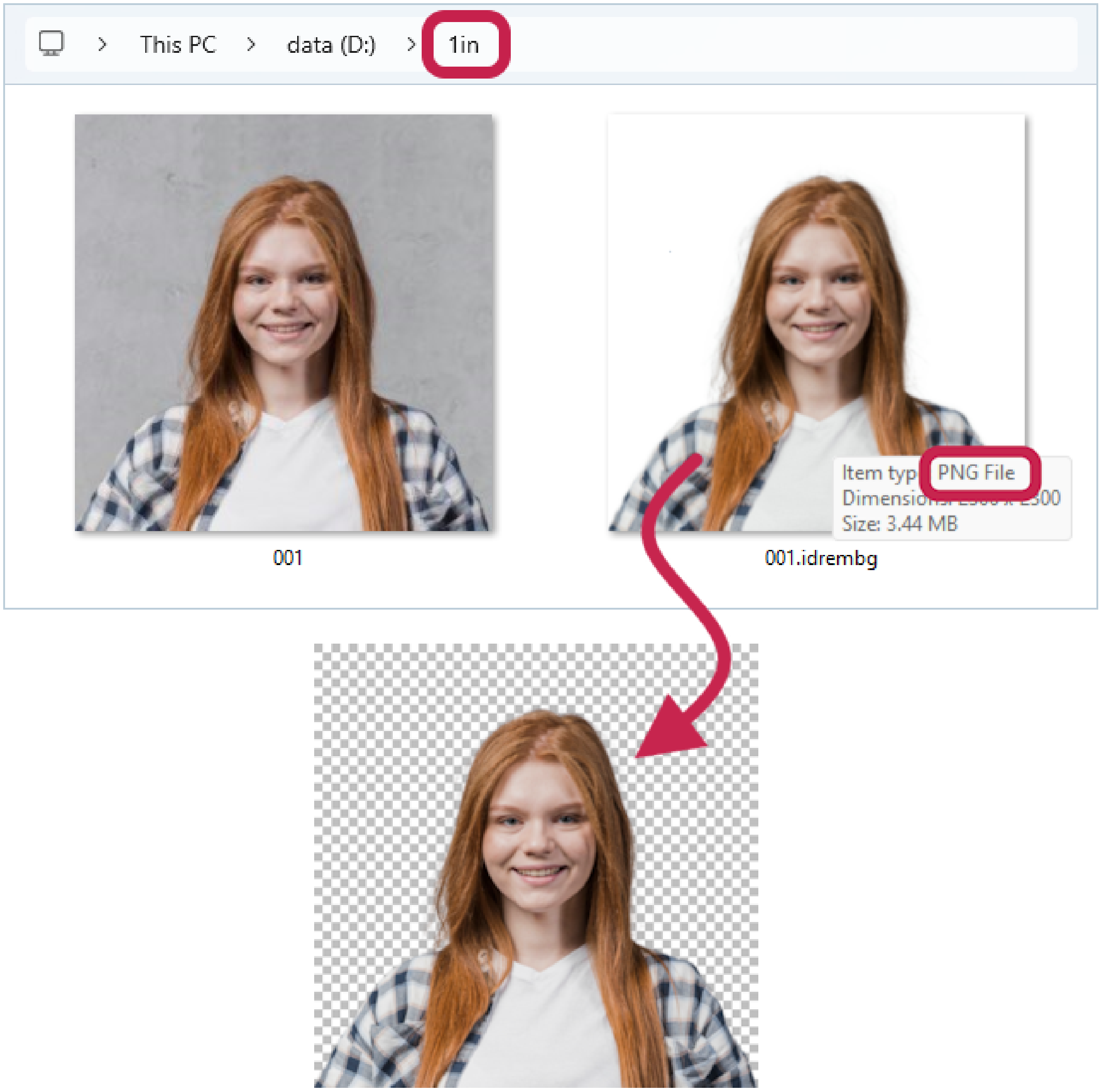
However, if source file is in a format like **JPEG** (**JPG**) that lacks transparency support, you will need to convert it to **PNG** or **TIFF** during processing. Use the -o (or --out-file) key to make this change.

An example with the original file in PNG format:

Let's process the original file in **PNG** 001.png format and create a output file with a transparent background. The transparent background is set using the -b key with the value transparent.

```
>idrembg -i "D:\1in\001.png" -b transparent
```

As a result, the file 001.idrembg.png with a fully transparent background will be saved in the D:\1in folder.



An example with the original file in JPEG format:

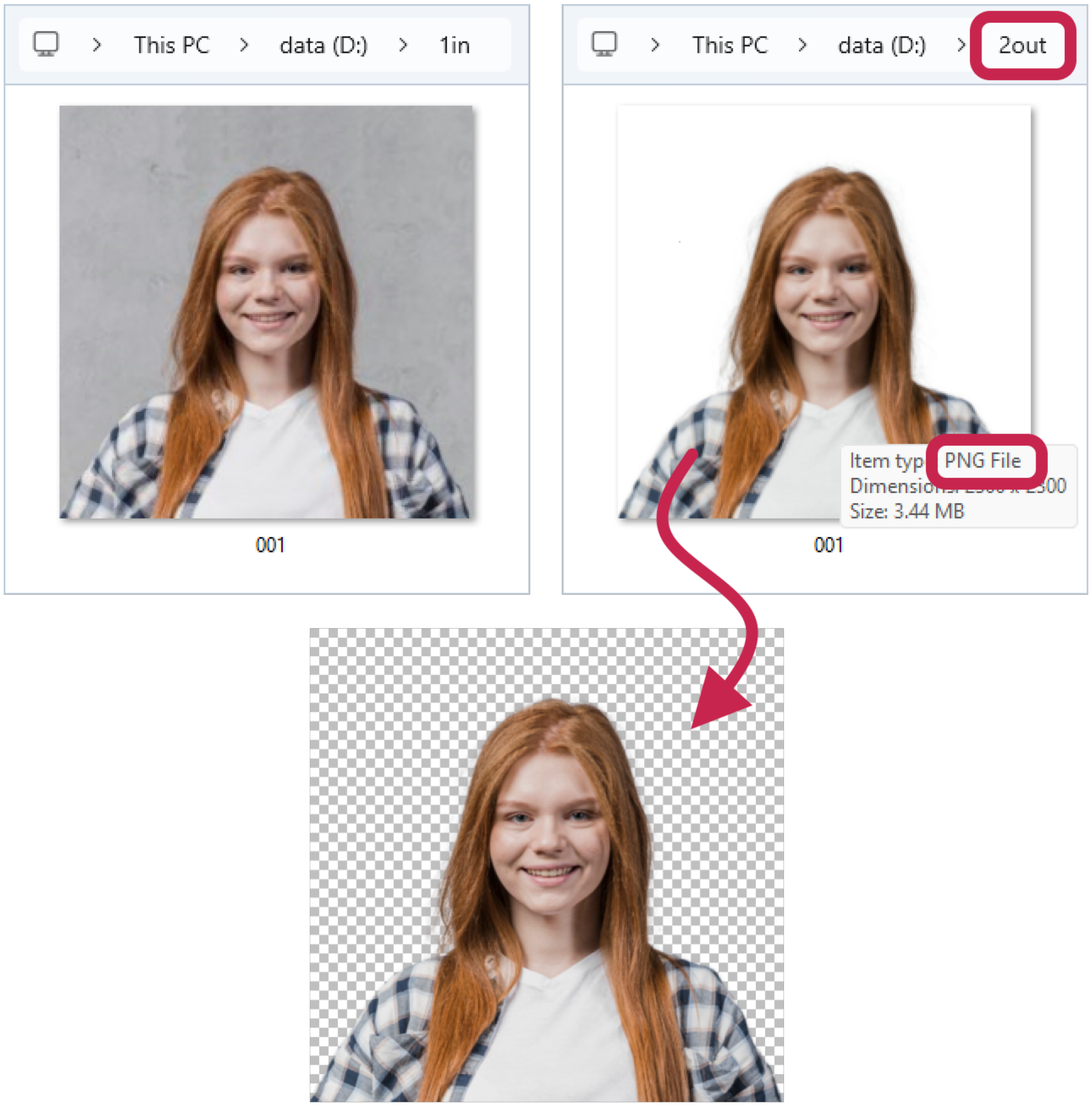
Let's process the original **JPEG** file 001.jpg and create a output **PNG** file with a transparent background.

The transparent background is set using the -b key with the value transparent.

The destination folder, name and format in which the processed file should be created will be specified in the -o key in the full name of the processed file - D:\2out\001.png.

```
>idrembg -i "D:\1in\001.jpg" -o "D:\2out\001.png" -b transparent
```

As a result, a **PNG** file 001.png with a completely transparent background will be saved to the D:\2out folder.



Background with Partial Transparency

To specify a background with partial transparency, you need to use **RGBA** (not **RGB**) or **HEX** color formats, which have a 4th value added - the degree of transparency.

In **RGBA**, transparency is specified by a number from 0 (transparent) to 1 (opaque): 0 corresponds to 0%, 0.5 - 50%, 0.8 - 80%, 1 - 100%. For example, in the color `rgba(230,230,230,230,0.5)` the transparency value is **0.5**.

In **HEX** format, transparency is expressed using hexadecimal numbers. For instance, in the color code #E6E6E6E680, the **80** at the end represents the transparency level, which indicates **50%** transparency.

Transparency codes for hexadecimal format are presented at the end of the document 38.

As previously mentioned, to achieve backgrounds with transparency, processed files should be created in PNG or TIFF formats 12.

An example with the original file in PNG format:

Let's process the original file in **PNG** format - 001.png and create a output file with a ***semi-transparent*** light blue background.

We will set the partial transparent background using the key -b with the value rgba(230,230,230,230,0.5) - in the first case and #E6E6E680 - in the second case.

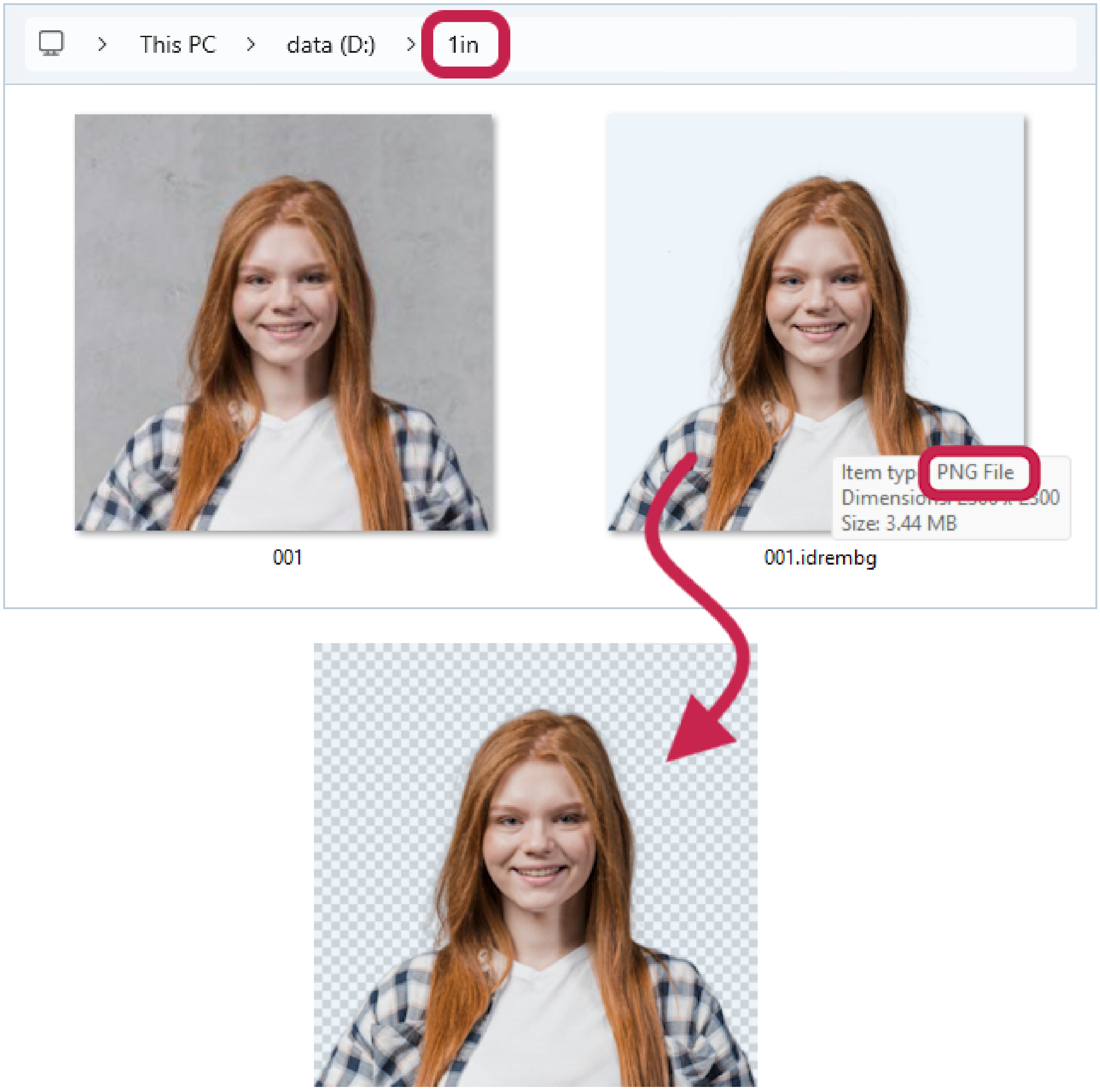
RGBA (transparency value - 0.5 corresponds to 50%)

```
>idrembg -i "D:\1in\001.png" -b rgba(230,230,230,0.5)
```

HEX (transparency value - 80 corresponds to 50%)

```
>idrembg -i "D:\1in\001.png" -b #E6E6E680
```

As a result,a **PNG** file 001.idrembg.png with a semi-transparent light blue background will be created in the D:\1in folder.



An example with the original file in JPEG format:

Let's process the original file 001.jpg and create a output file with a semi-transparent light blue background.

The partial transparent background is set using the -b key with the value rgba(230, 230, 230, 0.5) in the first instance and #E6E6E680 in the second, using the same values as in the previous example.

Because the source file, 001.jpg, is in **JPEG** format, which does not support transparency, it is necessary to specify the output format as PNG 12 when processing. To achieve this, include the -o key.

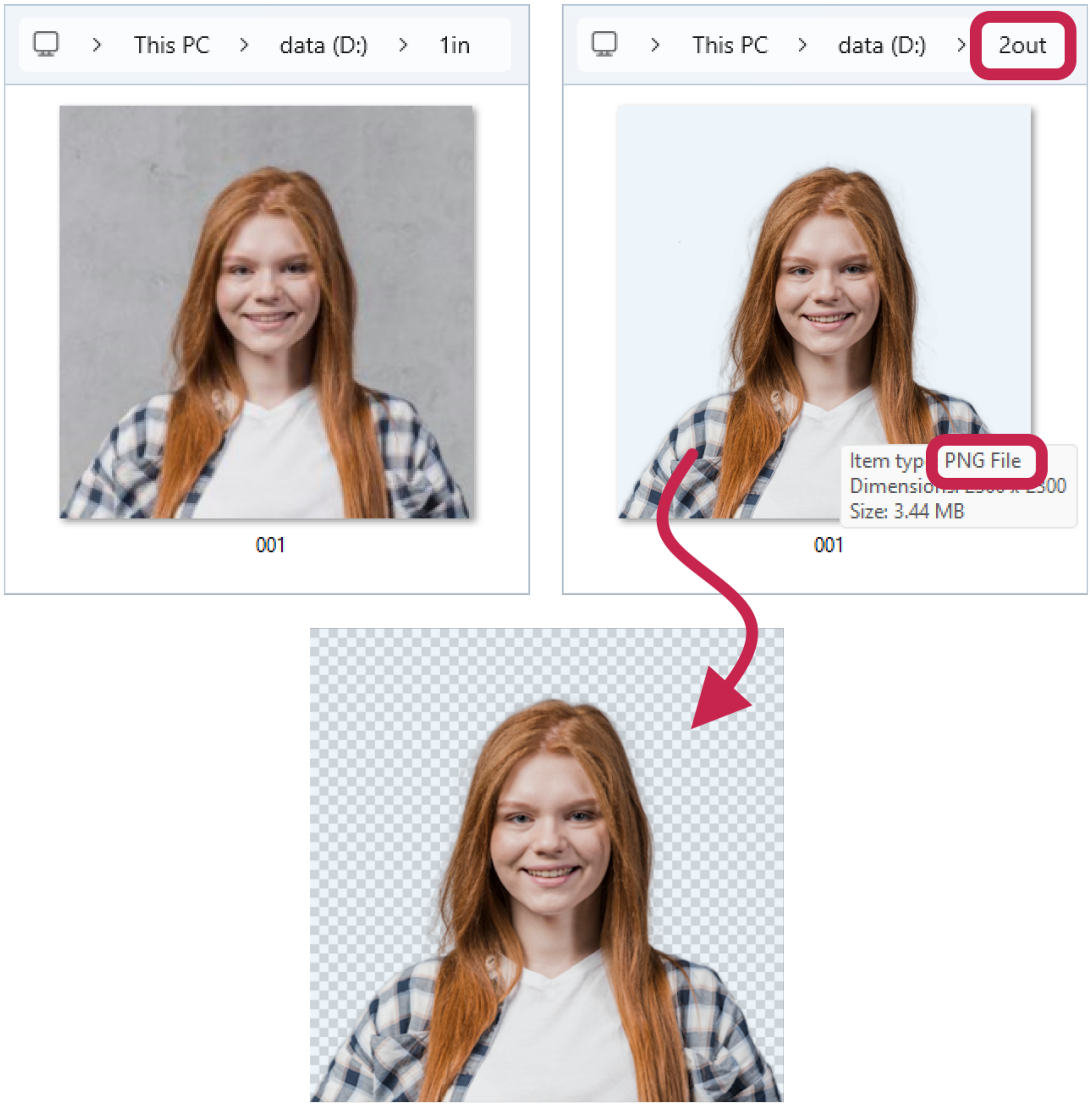
RGBA (transparency value - 0.5 corresponds to 50%)

```
>idrembg -i "D:\1in\001.jpg" -o "D:\2out\001.png" -b rgba(230,230,230,0.5)
```

HEX (transparency value - 80 corresponds to 50%)

```
>idrembg -i "D:\1in\001.jpg" -o "D:\2out\001.png" -b #E6E6E680
```

As a result, a **PNG** file 001.png with a semi-transparent light blue background will be created in the D:\2out folder.



Overwriting File

Files with the .idrembg suffix are overwritten automatically by the application.

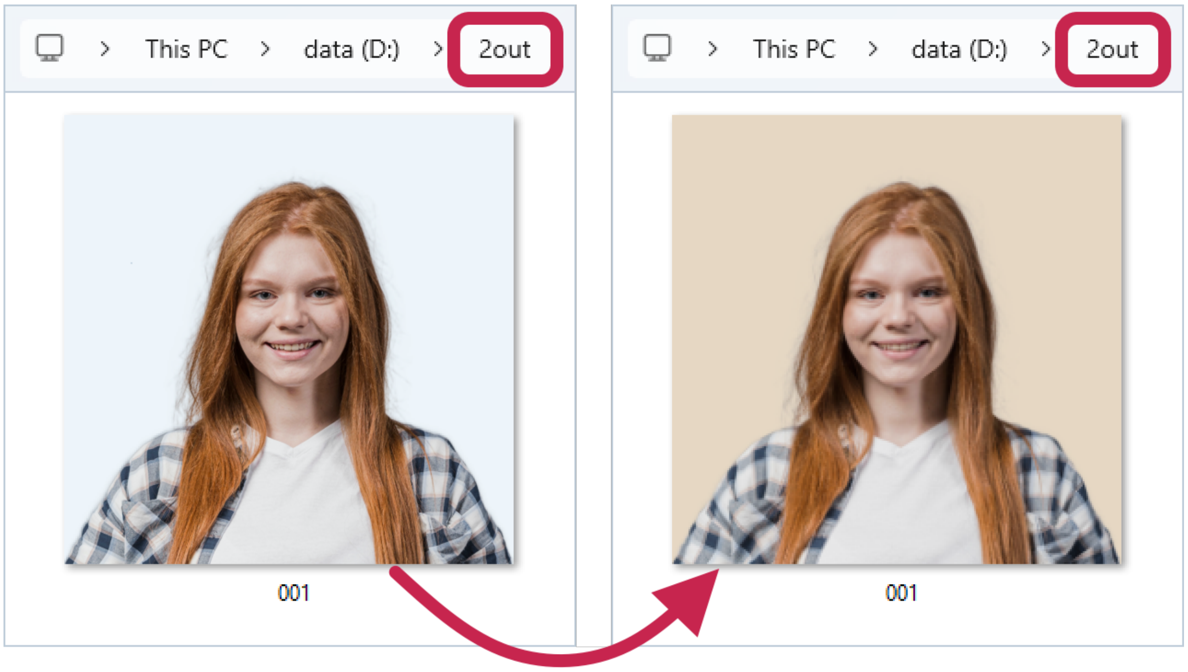
In other cases, use the --overwrite key.

Example:

The processed file D:\2out\001.jpg already exists - let's overwrite it with a new background color.

```
>idrembg -i "D:\1in\001.jpg" -o "D:\2out\001.jpg" -b rgb(230,215,195) --overwrite
```

As a result of executing this command line, the application will write the processed file D:\2out\001.jpg (with a sandy background) in place of the existing file with the same name (with a blue background).



Be careful when using the -i (--in-file) and --overwrite keys together without the -o key, as this will overwrite the original file!

Only use this combination if you are sure you will not need the source files in the future.

```
>idrembg -i "D:\1in\001.jpg" --overwrite
```

or

```
>idrembg --in-file "D:\1in\001.jpg" --overwrite
```

As a result of running such a command, the application will process the 001.jpg file and overwrite it. The original image will be overwritten without possibility of recovery!

Processing a Folder with Images

Processing a Folder with One Single Key

To avoid processing each image individually, you can use **IDPhoto Remove Background** to process an entire folder of images at once. All files in the folder with extensions such as **jpg**, **jpeg**, **png**, **webp**, **bmp**, **tif**, and **tiff** will be processed in a single operation. Simply use the --in-dir <path to the folder with source images> key to specify the path to the folder.

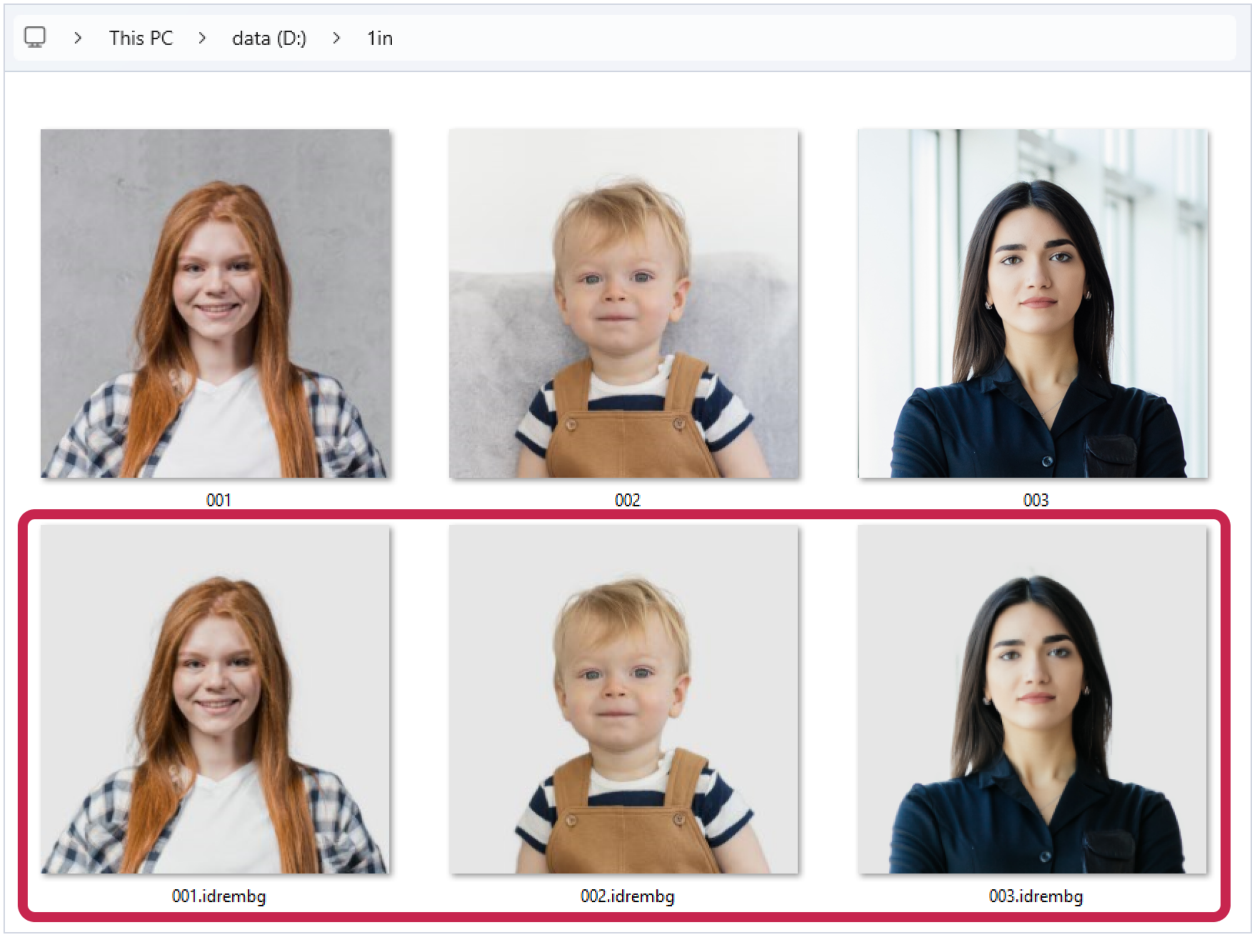
Example:

In the folder D:\1in there are images: 001.jpg, 002.jpg, 003.jpg. Let's start processing D:\1in with the --in-dir key:

```
>idrembg --in-dir "D:\1in"
```

This command line will create the processed files 001.idrembg.jpg, 002.idrembg.jpg, 003.idrembg.jpg in the D:\1in folder.

The background color of the processed photos will be set to light grey. This color is always set if no other color is set.



Saving Processed Images

The application can be directed to save the processed images to a specific location. Add the key --out-dir <folder path> to the command line to specify the desired folder path.

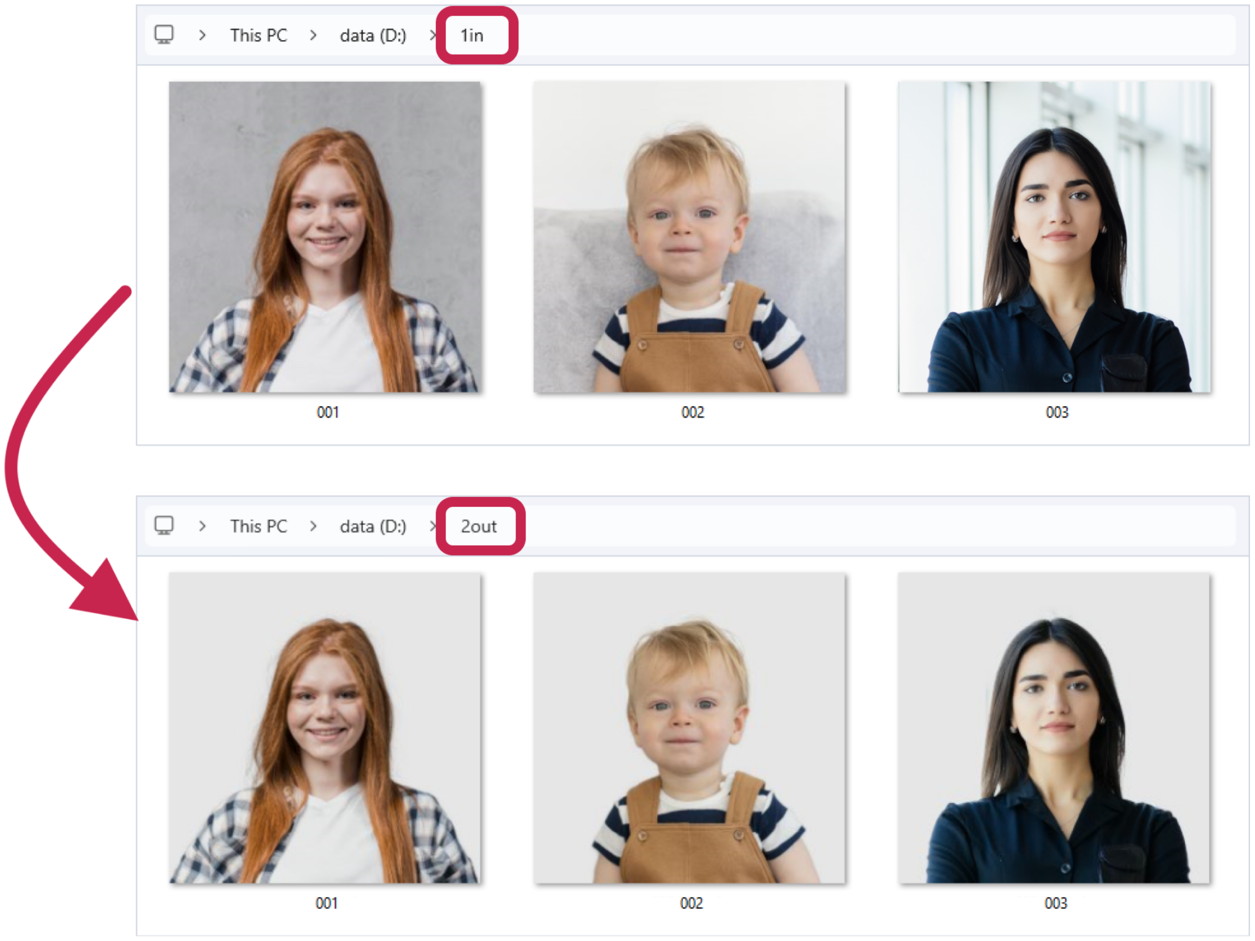
Example:

The images in the D:\1in folder are 001.jpg, 002.jpg, 003.jpg. Let's start processing the D:\1in folder and save the processed images to the D:\2out folder using the --out-dir key:

```
>idrembg --in-dir "D:\1in" --out-dir "D:\2out"
```

As a result of executing this command line, only the original files will remain in the D:\1in folder, and the processed files: 001.jpg, 002.jpg, 003.jpg will be saved in the D:\2out folder. Note that the names of the processed files in the D:\2out folder do not contain the .idrembg suffix.

The background color of the processed images will default to light grey. As mentioned earlier, this color is automatically applied if no other color is specified.



Background Color

To set a desired background color for the processed image, use the -b or --background <color> option.

Detailed information on handling background color in **IDPhoto Remove Background** can be found in the Single image processing -> Background Color 12 section.

Example:

In the folder D:\1in there are images: 001.jpg, 002.jpg, 003.jpg. Let's start processing the D:\1in folder and use the -b key to set the background color to light blue (#D0E3EF or rgb(208,227,239)).

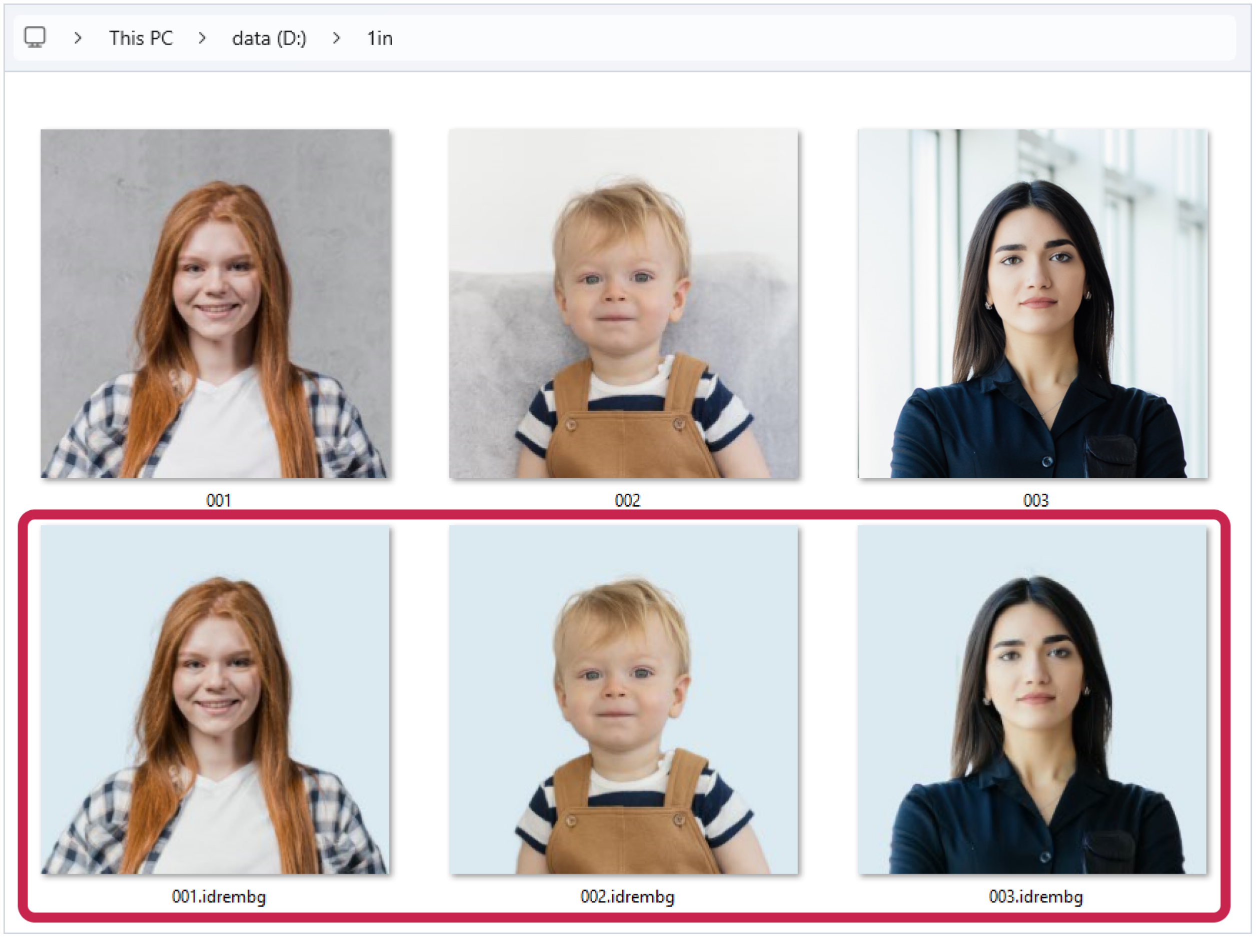
HEX

```
>idrembg --in-dir "D:\1in" -b #D0E3EF
```

RGB

```
>idrembg --in-dir "D:\1in" -b rgb(208,227,239)
```

As a result, the following processed images will be created in the D:\1in folder: 001.idrembg.jpg, 002.idrembg.jpg, 003.idrembg.jpg with light blue background color, as we specified in the -b key.



Transparent Background

To create processed images with a transparent background, the application must save them in a format that supports transparency, such as **PNG** or **TIFF**.

By default, the application preserves the format of the source file for the processed image. If original file is already in **PNG** or **TIFF** format, no further adjustments to file format are needed to maintain transparency.

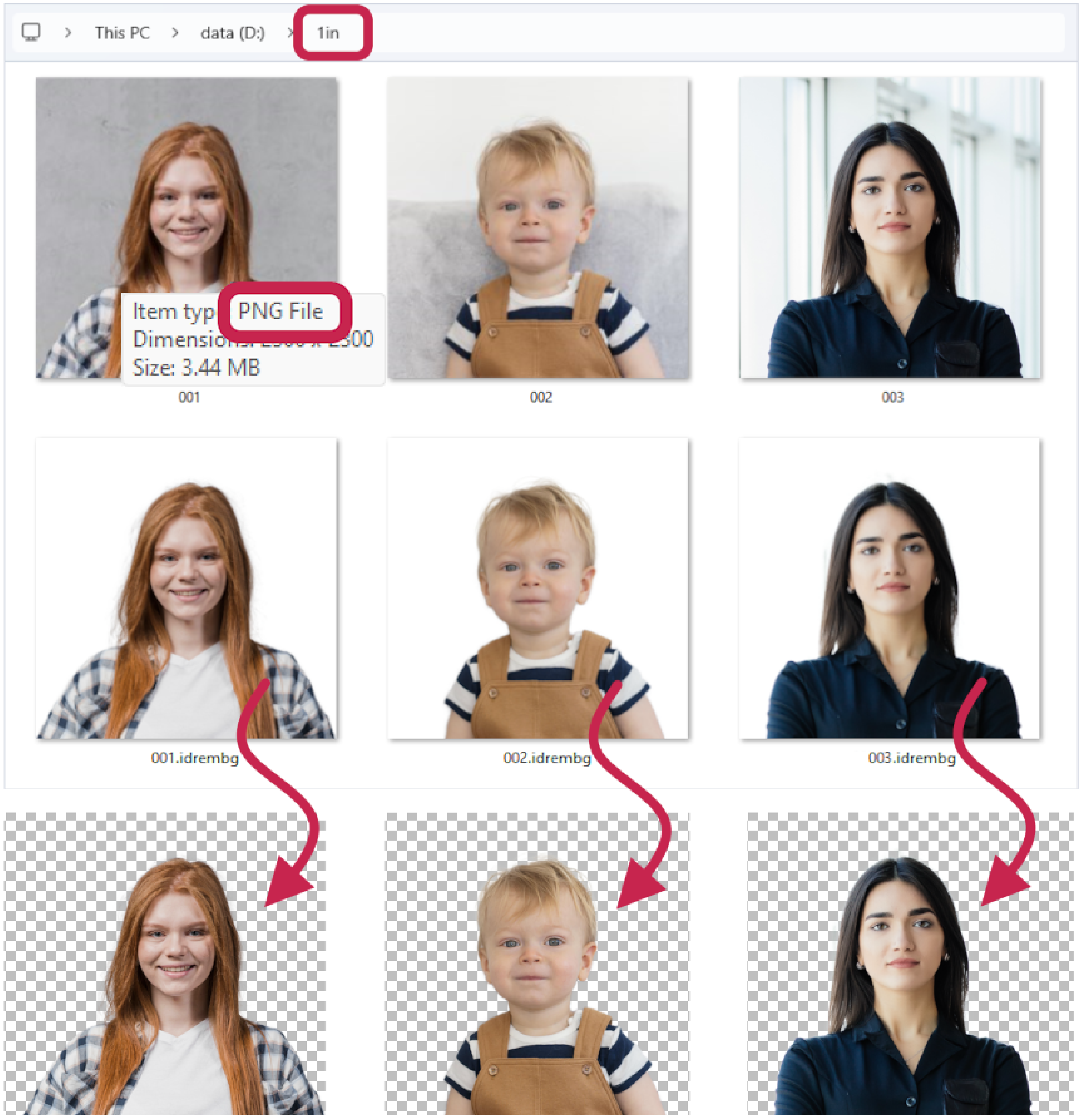
However, if source files is in a format like **JPEG** (**JPG**) that lacks transparency support, you will need to convert them to **PNG** or **TIFF** during processing. Use the --output-format key to make this change.

An example for a folder containing source files in PNG format:

The D:\1in folder contains images in PNG format: 001.png, 002.png, 003.png. Let's start processing the D:\1in folder and set a transparent background color using the -b key with the value transparent.

```
>idrembg --in-dir "D:\1in" -b transparent
```

As a result, the following processed images will be created in the D:\1in folder: 001.idrembg.png, 002.idrembg.png, 003.idrembg.png with a completely transparent background.



An example for a folder containing source files in JPEG format:

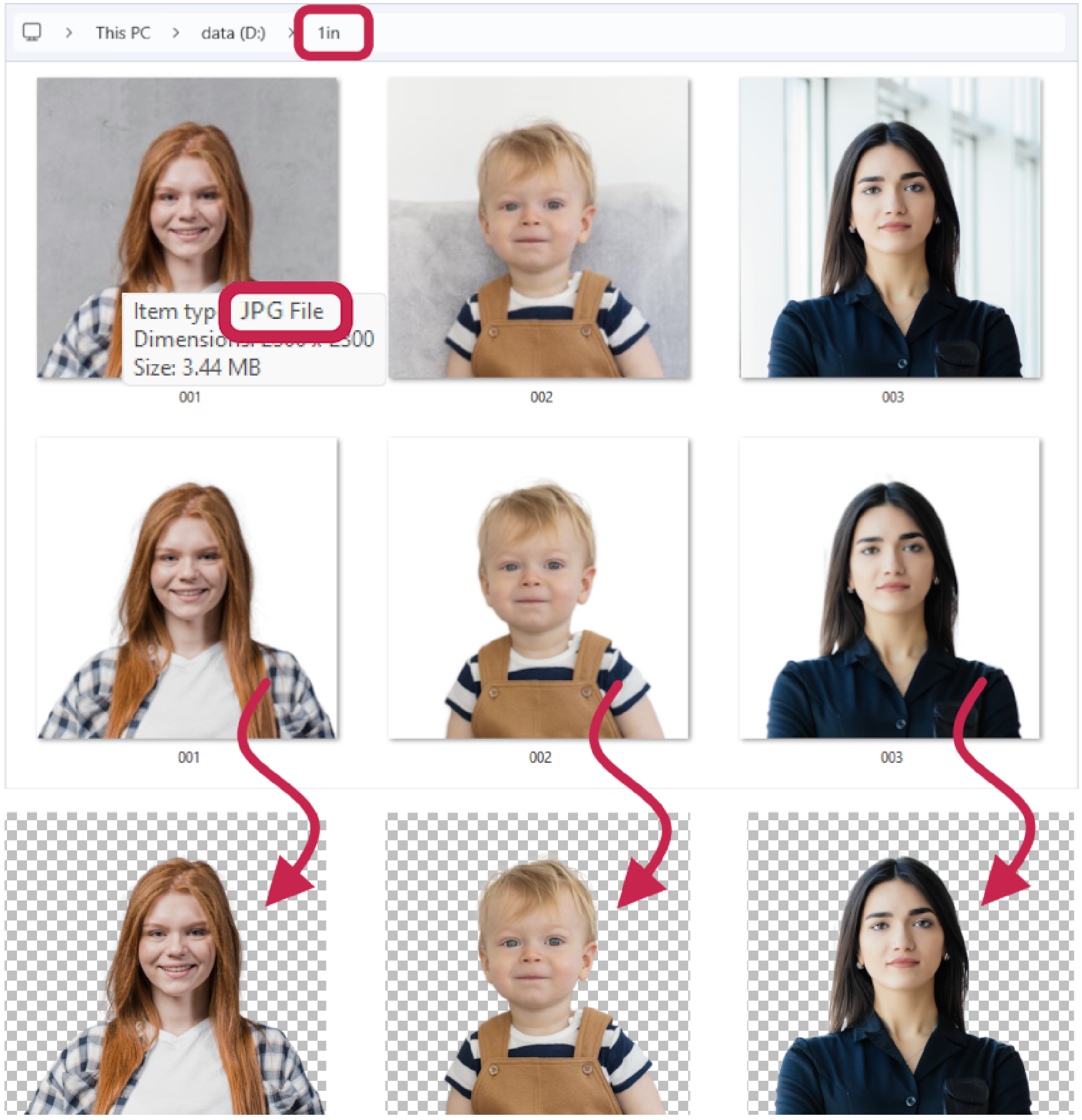
The D:\1in folder contains images in **JPEG** format: 001.jpg, 002.jpg, 003.jpg. Let's process the original **JPEG** files and create a output **PNG** files with a transparent background.

The transparent background is set using the -b key with the value transparent.

The format in which the processed files should be created is set using the --output-format key with the value png.

```
>idrembg --in-dir "D:\1in" -b transparent --output-format png
```

As a result, processed files in **PNG** format will be created in the D:\1in folder with source files in **JPG** format: 001.png, 002.png, 003.png with ***fully transparent*** background.



Background with Partial Transparency

To specify a background with partial transparency, you need to use **RGBA** (not **RGB**) or **HEX** color formats, which have a 4th value added - the degree of transparency.

In **RGBA**, transparency is specified by a number from 0 (transparent) to 1 (opaque): 0 corresponds to 0%, 0.5 - 50%, 0.8 - 80%, 1 - 100%. For example, in the color rgba(230,230,230,230,0.5) the transparency value is **0.5**.

In **HEX** format, transparency is expressed using hexadecimal numbers. For instance, in the color code #E6E6E6E680, the **80** at the end represents the transparency level, which indicates **50%** transparency.

Transparency codes for hexadecimal format are presented at the end of the document 38.

As previously mentioned, to achieve backgrounds with transparency, processed files should be created in PNG or TIFF formats 12.

An example for a folder containing source files in PNG format:

The D:\1in folder contains images in **PNG** format: 001.png, 002.png, 003.png. Let's start processing the D:\1in folder and set a ***semi-transparent*** light blue background color using the -b key.

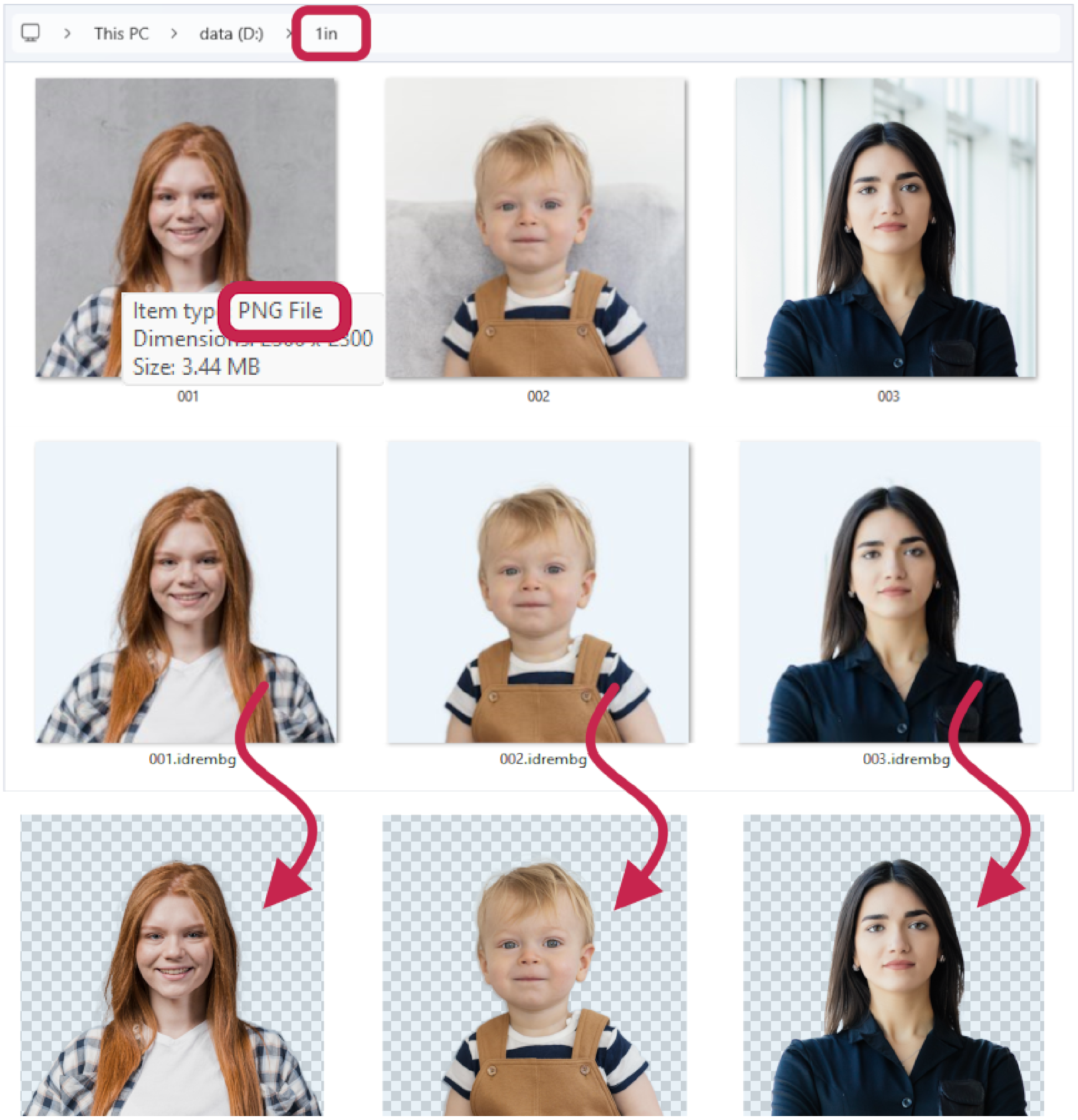
RGBA (transparency value - 0.5 corresponds to 50%)

```
>idrembg --in-dir "D:\1in" -b rgba(208,227,239,0.5)
```

HEX (transparency value - 80 corresponds to 50%)

```
>idrembg --in-dir "D:\1in" -b #D0E3EF
```

As a result, the following processed files will be created in the D:\1in folder: 001.idrembg.png, 002.idrembg.png, 003.idrembg.png with ***semi-transparent*** light blue background.



An example for a folder containing source files in JPEG format:

The D:\1in folder contains images in JPG format: 001.jpg, 002.jpg, 003.jpg. Let's start processing of the D:\1in folder and set a partial transparent light blue background color using the -b key. Add the --output-format key with the value **png** so that the processed files will be created in **PNG** format.

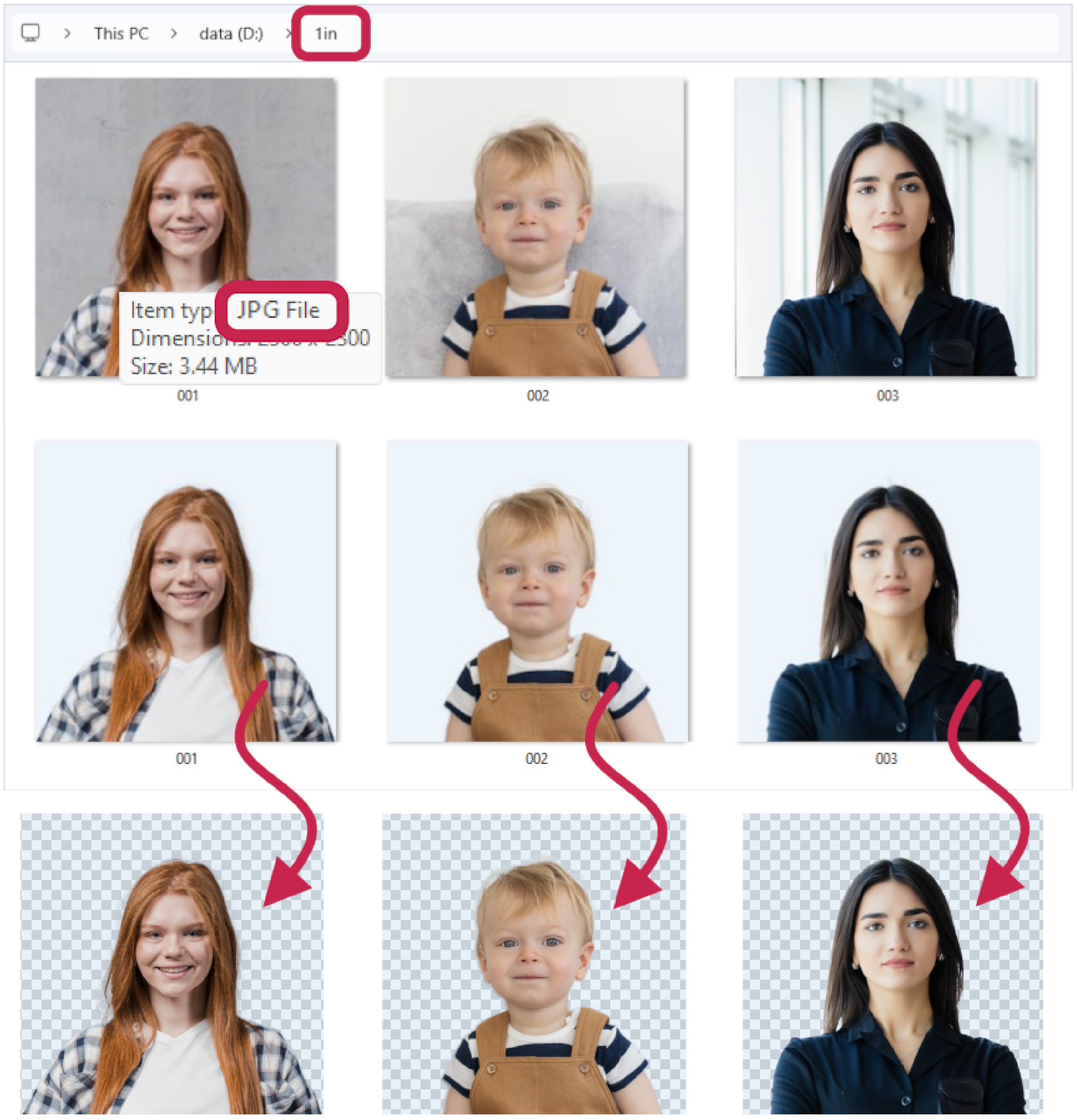
RGBA (transparency value of 0.5 corresponds to 50%)

```
>idrembg --in-dir "D:\1in " -b rgba(208,227,239,0.5) --output-format png
```

HEX (transparency value - 80 corresponds to 50%)

```
>idrembg --in-dir "D:\1in" -b #D0E3EF --output-format png
```

As a result, the processed files in **PNG** format (which we specified with the `--output-format` key) will be created in the `D:\1in` folder: `001.png`, `002.png`, `003.png` with a ***semi-transparent*** light blue background.



Overwriting Files in a Folder

Files with the .idrembg suffix are overwritten automatically by the application.

In other cases, use the --overwrite key.

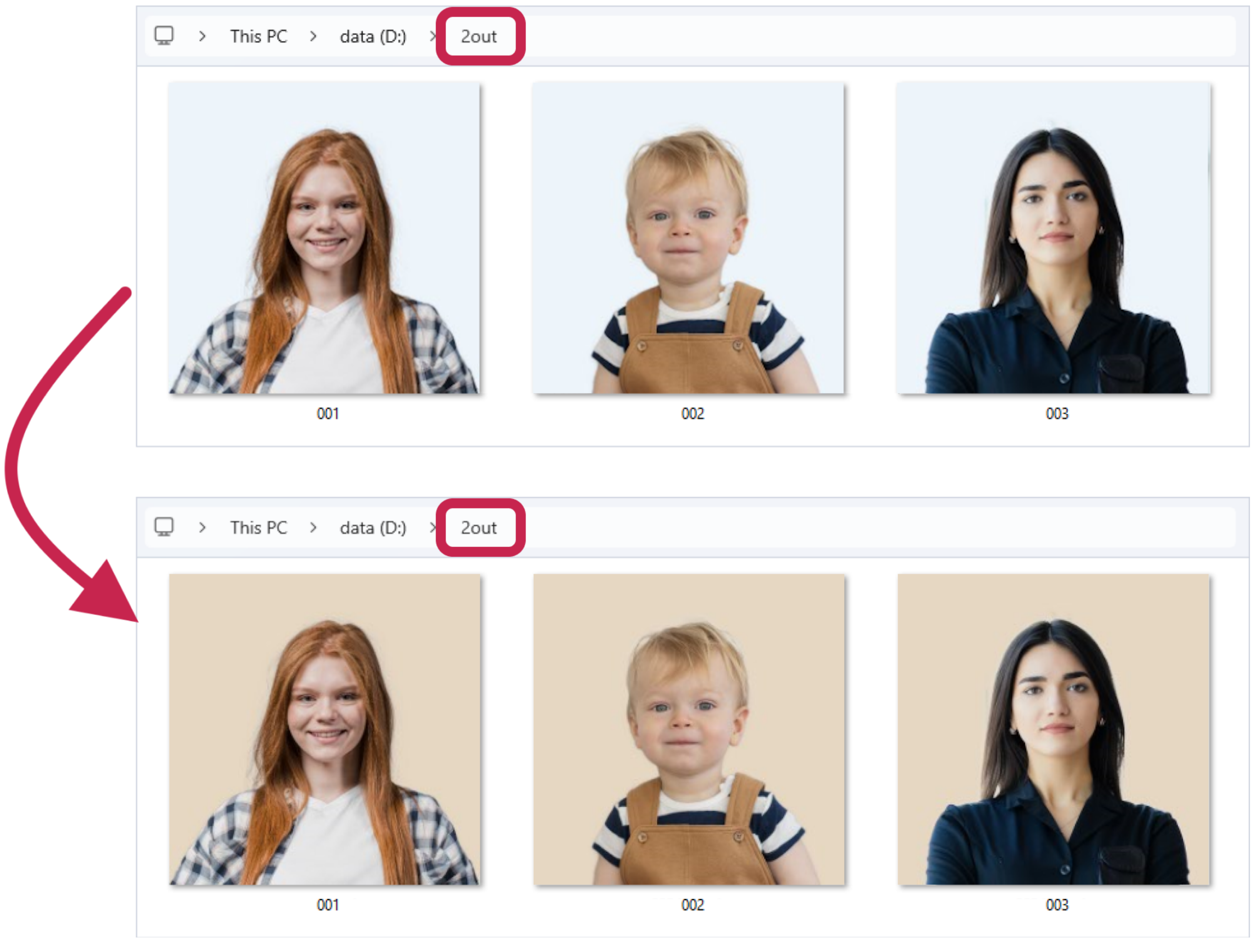
Example:

After processing the files in the D:\1in folder, we saved them to the D:\2out folder. The D:\2out folder now contains the processed files, which have a light grey background: 001.jpg, 002.jpg, and 003.jpg.

Then we needed to process the files in the D:\1in folder again, but with a different background color (white) and the previously processed files are no longer needed. To make the application overwrite the files in the D:\2out folder, apply the --overwrite key.

```
>idrembg --in-dir "D:\1in" --out-dir "D:\2out" -b rgb(230,215,195) --overwrite
```

As a result, the processed files will be saved to the D:\2out folder, and any existing files with the same name will be overwritten. In our example, all files in the D:\2out folder have been overwritten.
As shown in the screenshot, the processed images originally with a light blue background have been replaced with those featuring a sandy background.



Be careful when using --in-dir and --overwrite together without --out-dir, as this will overwrite the source files!

Only use this combination if you are sure you will not need the source files in the future.

Example:

```
>idrembg —in-dir "D:\1in" --overwrite
```

As a result of running this command, the application will process the D:\1in folder and write the processed files to it instead of the original files. The original images will be overwritten without the possibility of restoring them!

Good to Know

- To avoid writing the full path to the application in the command line, run **cmd** from the folder containing `idrembg.exe`.
  To do this, open the folder containing `idrembg.exe` in **Explorer**, type **cmd** in the path bar and press **Enter**.

- To avoid writing the full path to the application on the command line, regardless of **cmd** startup folder, write it to the environment variables.
  To do this:
  1. Press **Win+R** and type **sysdm.cpl**.
  2. In the **System Properties** dialogue box that opens, go to the **Advanced** tab and click **Environment Variables** at the bottom.
  3. In the window that opens, create a new user environment variable, specifying **Variable Name** - `idrembg`, **Variable Value** - the full path to `idrembg.exe`
  4. Close all dialogue windows by clicking **OK**
  5. Restart the computer

- Press **Ctrl+C** to stop the **cmd** command executing.

Support

If you have any questions, you can always contact our technical support team.

Please send an email to support@idphotocapture.com or use the feedback form on the site: https://idphotocapture.com/supports/

General Background Colors for ID photos

Color	Name	HEX	RGB
▨	*transparent*		
	white	#ffffff	(255,255,255)
⬛	*gray*	#808080	(128,128,128)
🟦	*skyblue*	#87ceeb	(135,206,235)
🟩	*limegreen*	#32cd32	(50,205,50)
⬜	*beige*	#f5f5dc	(245,245,220)
🟫	*tan*	#d2b48c	(210,180,140)
🟥	*firebrick*	#b22222	(178,34,34)
🟨	*lemonchiffon*	#fffacd	(255,250,205)
🟪	*mediumorchid*	#ba55d3	(186,85,211)
🟧	*peru*	#cd853f	(205,133,63)

HEX and RGBA Transparency Values

%	HEX	RGBA
100%	FF	1
90%	E6	0.9
80%	CC	0.8
75%	BF	0.75
70%	B3	0.7
60%	99	0.6
50%	80	0.5
40%	66	0.4
30%	4D	0.3
25%	40	0.25
20%	33	0.2
10%	1A	0.1