What is Image Upscaling?

Image upscaling is the process of increasing the resolution (i.e., the number of pixels) of an image while preserving or improving its visual quality.

Common use cases include:

- Enlarging small or low-resolution images (e.g. old photos, thumbnails) for printing or large displays.
- Enhancing quality for video streaming, gaming, or medical imaging.
- Improving clarity in computer vision tasks like object detection or segmentation.

In [1]:
```
import warnings
warnings.filterwarnings("ignore", category=UserWarning)
```

In [7]:
```
!pip install -q transformers accelerate safetensors diffusers realesrgan gfpgan
```

Why GANs Achieve the Best Results

GANs (Generative Adversarial Networks) have brought a breakthrough in image super-resolution by generating **new, realistic details** instead of just guessing.

How GANs Work:

- **Generator**: Creates high-resolution images from low-resolution input.
- **Discriminator**: Tries to distinguish between real and generated images.
- They compete — improving each other over time in a game-like training loop.

Importing Library

In [8]:
```
import os
import requests
import base64
import cv2
import torch
from torchvision import models, transforms
from PIL import Image
from gfpgan.utils import GFPGANer
from realesrgan.utils import RealESRGANer
from basicsr.archs.srvgg_arch import SRVGGNetCompact
from IPython.display import display
import os
import requests
from diffusers import DiffusionPipeline, StableDiffusionXLImg2ImgPipeline
from torchvision.transforms import ToTensor, Normalize, ConvertImageDtype
```

```
The cache for model files in Transformers v4.22.0 has been updated. Migrating you
r old cache. This is a one-time only operation. You can interrupt this and resume
the migration later on by calling `transformers.utils.move_cache()`.
0it [00:00, ?it/s]
```

Downloading the Model

In [9]:
```
model_urls = {
    'realesr-general-x4v3.pth': "https://github.com/xinntao/Real-ESRGAN/releases
    'GFPGANv1.4.pth': "https://github.com/TencentARC/GFPGAN/releases/download/v1
}

os.makedirs('weights', exist_ok=True)

def download_file(url, filename):
    response = requests.get(url, stream=True)
    if response.status_code == 200:
        with open(filename, 'wb') as f:
            for chunk in response.iter_content(chunk_size=1024):
                f.write(chunk)
        print(f"Downloaded {filename}")
    else:
        print(f"Failed to download {filename}. Status code: {response.status_cod

for filename, url in model_urls.items():
    file_path = os.path.join('weights', filename)
    if not os.path.exists(file_path):
        print(f"Downloading {filename}...")
        download_file(url, file_path)
    else:
        print(f"{filename} already exists. Skipping download.")
```

```
Downloading realesr-general-x4v3.pth...
Downloaded weights/realesr-general-x4v3.pth
Downloading GFPGANv1.4.pth...
Downloaded weights/GFPGANv1.4.pth
```

In [10]:
```
print(os.listdir('weights'))
```

```
['realesr-general-x4v3.pth', 'GFPGANv1.4.pth']
```

GFGAN vs Real-ESRGAN

Feature	GFGAN (Generative Facial Prior GAN)	Real-ESRGAN (Enhanced Super-Resolution GAN)
Purpose	Face image restoration	General image super-resolution & enhancement
Focus Area	Human faces only	All image types (faces, nature, text, etc.)
Architecture	GAN + Facial component priors	ESRGAN-based with U-Net discriminator
Input Types	Low-res, degraded face images	Real-world low-quality images (JPEG, noise, etc.)
Training Target	Facial detail accuracy (eyes, mouth, etc.)	Overall perceptual quality and texture realism
Pre-trained Models	Available for face restoration only	Available for multiple scales (x2, x4, etc.)

Feature	GFGAN (Generative Facial Prior GAN)	Real-ESRGAN (Enhanced Super-Resolution GAN)
Performance	Excels on faces, weak on general images	Strong performance across diverse image types
Use Case Example	Old portrait photo enhancement	Upscaling photos, anime, video frames, etc.

RealESRGAN

In [11]:
```
realesrgan_model_path = 'weights/realesr-general-x4v3.pth'

sr_model = SRVGGNetCompact(num_in_ch=3, num_out_ch=3, num_feat=64, num_conv=32,
half = True if torch.cuda.is_available() else False
realesrganer = RealESRGANer(scale=4, model_path=realesrgan_model_path, model=sr_

def upscale_image(image_path, output_path):
    img = cv2.imread(image_path, cv2.IMREAD_UNCHANGED)
    output, _ = realesrganer.enhance(img, outscale=4)
    cv2.imwrite(output_path, output)
    return output
```

GFPGAN

In [12]:
```
gfpgan_model_path = 'weights/GFPGANv1.4.pth'

face_enhancer = GFPGANer(model_path=gfpgan_model_path, upscale=10, arch='clean',

# Function to enhance image with GFPGAN
def enhance_faces(image_path, output_path):
    img = cv2.imread(image_path, cv2.IMREAD_UNCHANGED)
    _, _, img_enhanced = face_enhancer.enhance(img, has_aligned=False, only_cent
    cv2.imwrite(output_path, img_enhanced)
    return img_enhanced
```

```
Downloading: "https://github.com/xinntao/facexlib/releases/download/v0.1.0/detect
ion_Resnet50_Final.pth" to /kaggle/working/gfpgan/weights/detection_Resnet50_Fina
l.pth

100%|██████████| 104M/104M [00:00<00:00, 138MB/s]
Downloading: "https://github.com/xinntao/facexlib/releases/download/v0.2.2/parsin
g_parsenet.pth" to /kaggle/working/gfpgan/weights/parsing_parsenet.pth

100%|██████████| 81.4M/81.4M [00:00<00:00, 271MB/s]
```

Initial image

In [15]:
```
initial_image_path = '/kaggle/input/old-photos/old_photo_01.jpg'

# Load the image with PIL
photo = Image.open(initial_image_path)
display(photo.resize((800, 800), Image.LANCZOS))
```

Enhance and upscale

In [16]:
```
enhanced_faces_path = "/kaggle/working/enhanced_faces.jpg"

try:
    enhance_faces(initial_image_path, enhanced_faces_path)
    enhanced_image_to_display = Image.open(enhanced_faces_path)
    display(enhanced_image_to_display)

except Exception as err:
    print(f"An error occurred during face enhancement: {err}")
```

```
/opt/conda/lib/python3.10/site-packages/PIL/Image.py:3176: DecompressionBombWarni
ng: Image size (94960000 pixels) exceeds limit of 89478485 pixels, could be decom
pression bomb DOS attack.
  warnings.warn(
```

In [ ]: