最终的效果图如下：

想想怎么推广网站或者分享网站的文章呢？想想前些日子帮原同事简单生成了一批保存人员代码的二维码，想着我可以为自己的网站生成二维码用以推广啊，和先前一样使用 `qrcode` 库来生成二维码图片。 下面是简单生成的示例代码，使用 Python 生成二维码图片并保存为 PNG 文件。

如果尚未安装 `qrcode` 库，可以通过以下命令安装它：

```
pip install qrcode[pil]
```

1. 简单示例

```
import qrcode

# 定义二维码内容
data = "https://www.example.com"

# 创建QRCode对象
qr = qrcode.QRCode(
    version=1,  # 控制二维码大小，1是21x21的大小
    error_correction=qrcode.constants.ERROR_CORRECT_L,  # 容错率
    box_size=10,  # 每个二维码格子的像素大小
    border=4,  # 边框宽度
)

# 将数据添加到二维码中
qr.add_data(data)
qr.make(fit=True)

# 创建二维码图片
img = qr.make_image(fill='black', back_color='white')

# 保存二维码图片
img.save("qrcode.png")
```

你可以根据需要修改二维码的内容、样式或保存路径。

2. 将网站logo加入到二维码正中央

使用`PIL.Image` 模块中的 `Image.LANCZOS` 作为图像缩放的过滤器

```
import qrcode
from PIL import Image

# 生成二维码
data = "https://www.example.com"  # 二维码内容
```

```
qr = qrcode.QRCode(
    version=1,  # 控制二维码大小
    error_correction=qrcode.constants.ERROR_CORRECT_H,  # 提高容错率
    box_size=10,  # 每个二维码格子的像素大小
    border=4,  # 边框宽度
)
qr.add_data(data)
qr.make(fit=True)

# 生成二维码图片
qr_img = qr.make_image(fill='black', back_color='white').convert('RGB')

# 打开并调整网站logo大小
logo_path = "/content/favicon.ico"
logo = Image.open(logo_path)

# 根据二维码的大小调整logo的大小
qr_width, qr_height = qr_img.size
logo_size = int(qr_width * 0.2)  # logo的大小为二维码的20%
logo = logo.resize((logo_size, logo_size), Image.LANCZOS)  # 使用 LANCZOS 替代 ANTIALIAS

# 计算logo放置在二维码中央的位置
logo_pos = ((qr_width - logo_size) // 2, (qr_height - logo_size) // 2)

# 将logo粘贴到二维码上
qr_img.paste(logo, logo_pos, mask=logo)

# 保存二维码图片
qr_img.save("qrcode_with_logo.png")

# 显示生成的二维码
qr_img.show()
```

3. 以Logo为背景图生成二维码

logo图颜色太过鲜艳显眼，可以同时降低它的透明度并添加一个半透明的蒙版来增强二维码的可读性。下面是结合了调节背景图透明度和添加半透明蒙版的代码：

```
import qrcode
from PIL import Image, ImageEnhance

# 生成二维码
data = "https://www.example.com"  # 二维码内容
qr = qrcode.QRCode(
    version=1,  # 控制二维码大小
    error_correction=qrcode.constants.ERROR_CORRECT_H,  # 提高容错率
    box_size=20,  # 每个二维码格子的像素大小，以适应背景图
    border=0,  # 去除二维码边框
)
qr.add_data(data)
qr.make(fit=True)

# 生成二维码图片，并转换为 RGBA 模式，支持透明背景
qr_img = qr.make_image(fill='black', back_color='white').convert('RGBA')

# 打开logo图片，并调整大小为200x200
logo_path = "/content/favicon.ico"
logo = Image.open(logo_path).convert("RGBA")
```

```
logo = logo.resize((200, 200), Image.LANCZOS)

# 调节背景图片的透明度
alpha = 0.5  # 设置透明度，0表示完全透明，1表示不透明
enhancer = ImageEnhance.Brightness(logo)
logo = enhancer.enhance(alpha)

# 创建一个半透明的白色蒙版
mask = Image.new('RGBA', qr_img.size, (255, 255, 255, 128))  # 半透明白色

# 调整二维码图片大小为200x200
qr_img = qr_img.resize((200, 200), Image.LANCZOS)

# 将二维码的白色部分替换为透明
datas = qr_img.getdata()
new_data = []
for item in datas:
    # 将白色（255, 255, 255）替换为透明
    if item[:3] == (255, 255, 255):  # RGB是白色时
        new_data.append((255, 255, 255, 0))  # 透明
    else:
        new_data.append(item)

qr_img.putdata(new_data)

# 在背景图上先叠加半透明蒙版
logo.paste(mask, (0, 0), mask)

# 然后将二维码叠加到背景图上
logo.paste(qr_img, (0, 0), qr_img)

# 保存并显示结果图片
logo.save("qrcode_with_combined_effects.png")
logo.show()
```

关键步骤

1. **调节背景图的透明度**：通过 `ImageEnhance.Brightness` 降低背景图的亮度，`alpha` 控制透明度（在这里设置为 0.5，可以根据需求调整）。
2. **添加半透明蒙版**：创建一个白色的半透明蒙版，将其叠加在背景图上，使背景颜色更加柔和，增强二维码的可读性。
3. **调整二维码透明度**：将二维码的白色部分替换为透明，以使背景图透过透明区域显示。
4. **将二维码叠加到背景图上**：在调节过透明度和添加蒙版的背景图上叠加二维码，使两者更加协调。

4. 进一步将背景图换成动态图（GIF类型）

将动态 GIF 图像作为二维码的背景并保持其动态效果，需要逐帧处理 GIF 图像。对于每一帧，使用以上类似的操作来处理背景透明度、添加蒙版，并将二维码叠加在每一帧上，最后将这些帧合并为新的 GIF 文件。可调节大小，在处理 GIF 图像的过程中调整背景和二维码的大小。可以在每一帧上调整 GIF 背景图像和二维码的尺寸，使它们匹配，按需要进行缩放。 可以通过调整每一帧和二维码的大小来实现这个功能。以下是它的代码：

```
import qrcode
from PIL import Image, ImageEnhance, ImageSequence

# 生成二维码
data = "https://www.example.com"  # 二维码内容
qr = qrcode.QRCode(
    version=1,  # 控制二维码大小
```

```
    error_correction=qrcode.constants.ERROR_CORRECT_H,  # 提高容错率
    box_size=10,  # 每个二维码格子的像素大小
    border=0,  # 去除二维码边框
)
qr.add_data(data)
qr.make(fit=True)

# 生成二维码图片，并转换为 RGBA 模式，支持透明背景
qr_img = qr.make_image(fill='black', back_color='white').convert('RGBA')

# 将二维码的白色部分替换为透明
datas = qr_img.getdata()
new_data = []
for item in datas:
    # 将白色（255, 255, 255）替换为透明
    if item[:3] == (255, 255, 255):  # RGB是白色时
        new_data.append((255, 255, 255, 0))  # 透明
    else:
        new_data.append(item)

qr_img.putdata(new_data)

# 打开动态GIF图片
gif_path = "/content/your_dynamic_background.gif"  # 替换为你的GIF路径
gif = Image.open(gif_path)

# 设置需要的背景图和二维码的尺寸
new_size = (200, 200)  # 目标大小，可以根据需要调整

# 调整二维码的大小
qr_img = qr_img.resize(new_size, Image.LANCZOS)

# 用于存储处理后的帧
frames = []

# 遍历GIF的每一帧
for frame in ImageSequence.Iterator(gif):
    # 将每一帧转换为 RGBA 模式
    frame = frame.convert('RGBA')

    # 调节背景图片的透明度
    alpha = 0.5  # 设置透明度，0表示完全透明，1表示不透明
    enhancer = ImageEnhance.Brightness(frame)
    frame = enhancer.enhance(alpha)

    # 调整每一帧的大小与二维码一致
    frame = frame.resize(new_size, Image.LANCZOS)

    # 创建一个半透明的白色蒙版
    mask = Image.new('RGBA', qr_img.size, (255, 255, 255, 128))  # 半透明白色

    # 将蒙版叠加到帧上
    frame.paste(mask, (0, 0), mask)

    # 将二维码叠加到帧上
    frame.paste(qr_img, (0, 0), qr_img)

    # 添加处理后的帧到frames列表中
    frames.append(frame)
```

```
# 保存结果为新的GIF，保留原GIF的帧速率和循环设置
frames[0].save(
    "qrcode_with_dynamic_background_resized.gif",
    save_all=True,
    append_images=frames[1:],  # 添加其他帧
    duration=gif.info['duration'],  # 使用原始的帧持续时间
    loop=gif.info.get('loop', 0)  # 保留循环设置
)
```

调节大小的步骤

1. **二维码大小调整**：使用 `qr_img.resize(new_size, Image.LANCZOS)` 通过 `new_size` 参数设置二维码的目标大小。例如，`new_size = (200, 200)` 调整为 200x200 像素。
2. **背景图帧大小调整**：在每一帧上通过 `frame.resize(new_size, Image.LANCZOS)` 同样将 GIF 背景调整到目标大小 `new_size`。
3. **蒙版大小调整**：蒙版的大小与二维码和背景一致，通过 `mask = Image.new('RGBA', qr_img.size, ...)` 创建与二维码大小相同的蒙版。

结果

生成的动态 GIF 将保持背景的动态效果，并且二维码和背景图都将按照指定的尺寸进行缩放。如果希望二维码比背景稍小或者更大，可以分别调整二维码和背景的 `resize` 参数。