

显示来自网上的图片

显示图片是大多数移动应用程序的基础。Flutter提供了 [Image](#) Widget来显示不同类型的图片。

为了处理来自URL的图像，请使用[Image.network](#)构造函数。

```
new Image.network(  
  'https://raw.githubusercontent.com/flutter/website/master/_includes/code/layout/lakes/images/lake.jpg',  
)
```

Bonus: GIF动画

[Image](#) Widget的一个惊艳的功能是：支持GIF动画！

```
new Image.network(  
  'https://github.com/flutter/plugins/raw/master/packages/video_player/doc/demo_ipod.gif?raw=true',  
);
```

占位图和缓存

[Image.network](#)默认不能处理一些高级功能，例如在下载完图片后加载或缓存图片到设备中后，使图片渐隐渐显。要实现这种功能，请参阅以下内容：

- [用占位符淡入图片](#)
- [使用缓存图片](#)

完整的例子

```
import 'package:flutter/material.dart';  
  
void main() => runApp(new MyApp());  
  
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    var title = 'Web Images';  
  
    return new MaterialApp(  
      title: title,  
      home: new Scaffold(  
        appBar: new AppBar(  
          title: new Text(title),  
        ),  
        body: new Image.network(  
          'https://github.com/flutter/website/blob/master/_includes/code/layout/lakes/images/lake.jpg?raw=true',  
        ),  
      ),  
    );  
  }  
}
```

用占位符淡入图片

当使用默认 `Image` widget 显示图片时，您可能会注意到它们在加载完成后会直接显示到屏幕上。这可能会让用户产生视觉突兀。

相反，如果你最初显示一个占位符，然后在图像加载完显示时淡入，那么它会不会更好？我们可以使用 `FadeInImage` 来达到这个目的！

`FadeInImage` 适用于任何类型的图片：内存、本地 `Asset` 或来自网上的图片。

在这个例子中，我们将使用 `transparent_image` 包作为一个简单的透明占位图。您也可以考虑按照 [Assets 和图片](#) 指南使用本地资源来做占位图。

```
new FadeInImage.memoryNetwork(  
  placeholder: kTransparentImage,  
  image: 'https://github.com/flutter/website/blob/master/_includes/code/layout/lakes/images/lake.jpg?  
raw=true',  
);
```

完整的例子

```
import 'package:flutter/material.dart';  
import 'package:transparent_image/transparent_image.dart';
```

```
void main() {  
  runApp(new MyApp());  
}
```

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    final title = 'Fade in images';
```

```
    return new MaterialApp(  
      title: title,  
      home: new Scaffold(  
        appBar: new AppBar(  
          title: new Text(title),  
        ),  
        body: new Stack(  
          children: <Widget>[  
            new Center(child: new CircularProgressIndicator()),  
            new Center(  
              child: new FadeInImage.memoryNetwork(  
                placeholder: kTransparentImage,  
                image:
```

```
'https://github.com/flutter/website/blob/master/_includes/code/layout/lakes/images/lake.jpg?  
raw=true',  
            ),  
          ],  
        ),  
      ),  
    );
```

```

    ],
  ),
),
);
}

```

使用缓存图片

在某些情况下，在从网上下载图片后缓存图片可能会很方便，以便它们可以脱机使用。为此，我们可以使用[cached_network_image](#)包来达到目的。

除了缓存之外，`cached_image_network`包在加载时还支持占位符和淡入淡出图片！

```

new CachedNetworkImage(
  imageUrl:
    'https://github.com/flutter/website/blob/master/_includes/code/layout/lakes/images/lake.jpg?raw=true',
);

```

添加一个占位符

`cached_network_image`包允许我们使用任何Widget作为占位符！在这个例子中，我们将在图片加载时显示一个进度圈。

```

new CachedNetworkImage(
  placeholder: new CircularProgressIndicator(),
  imageUrl:
    'https://github.com/flutter/website/blob/master/_includes/code/layout/lakes/images/lake.jpg?raw=true',
);

```

完整的例子

```

import 'package:flutter/material.dart';
import 'package:cached_network_image/cached_network_image.dart';

```

```

void main() {
  runApp(new MyApp());
}

```

```

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    final title = 'Cached Images';

    return new MaterialApp(
      title: title,
      home: new Scaffold(
        appBar: new AppBar(
          title: new Text(title),
        ),
        body: new Center(
          child: new CachedNetworkImage(
            placeholder: new CircularProgressIndicator(),

```

```
        imageUrl:
'https://github.com/flutter/website/blob/master/_includes/code/layout/lakes/images/lake.jpg?raw=true',
    ),
    ),
    ),
    );
}
```