1.10 Анимация

Анимация

# Неявная анимация

С помощью неявной анимации можно анимировать свойство элемента управления, задав целевое значение; при изменении этого целевого значения элемент управления анимирует свойство от старого значения к новому. Анимация создает интерполированные значения между старым и новым значением в течение заданного времени. По умолчанию анимация линейно увеличивает значение, однако к ней можно применить кривую, которая изменяет значение в соответствии с

заданной кривой. Например, кривая быстро увеличивает значение

easeOutCubic

анимации в начале анимации, а затем замедляет его до достижения целевого значения:

Каждый элемент управления предоставляет ряд свойств animate\_{something} , описанных ниже, для обеспечения неявной анимации его внешнего вида:

* (анимация прозрачности)

animate\_opacity

* (анимация вращения)

animate\_rotation

* (анимация масштабирования)

animate\_scale

* (анимация смещения)

animate\_offset

* animate\_position (анимация позиционирования)
* (Container) (анимация контейнера)

animate

Свойство может иметь одно из следующих значений:

animate\_\*

* Экземпляр класса - позволяет настроить

animation.Animation

длительность (в миллисекундах) и кривую анимации, например,

|  |  |  |  |
| --- | --- | --- | --- |
|  | animate\_rotation=ft.animation.Animation(duration=300, | | |
| curve="bounceOut") | | | . По умолчанию - линейная. |
| int value | | - включает анимацию с указанной длительностью в | |

миллисекундах и линейной кривой.

* - включает анимацию с длительностью 1000 миллисекунд и линейной кривой.

bool value

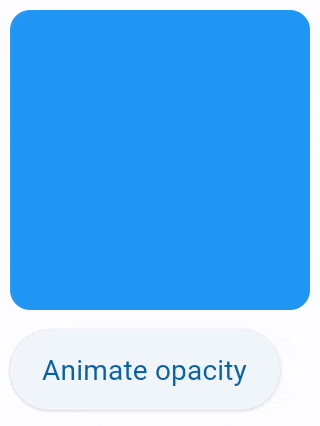
# Анимация непрозрачности

Установка параметра number или экземпля анимировать свойство

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| animate\_opacity | | |  | |
| р класса | | animation.Animation | | |
|  | Control.opacity | | | . |

элемента управления в значение True,

позволяет неявно



import flet as ft

def main(page: ft.Page):

c = ft.Container(

width=150,

height=150,

bgcolor="blue",

border\_radius=10,

animate\_opacity=300,

)

def animate\_opacity(e):

c.opacity = 0 if c.opacity == 1 else 1

c.update()

page.add(

c,

ft.ElevatedButton(

"Animate opacity", # Анимация непрозрачности

on\_click=animate\_opacity,

),

)

ft.app(target=main)

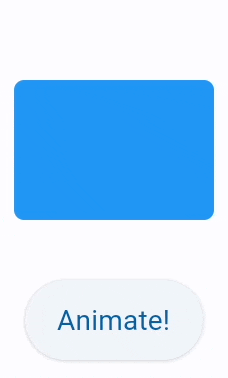
# Анимация вращения

Установка параметра number или экземпля анимировать свойство

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| animate\_rotation | | | |  |
| р класса | | animation.Animation | | |
|  | Control.rotate | | . | |

элемента управления в значение True ,

позволяет неявно



from math import pi

import flet as ft

def main(page: ft.Page):

c = ft.Container(

width=100,

height=70,

bgcolor="blue",

border\_radius=5,

rotate=ft.transform.Rotate(0, alignment=ft.alignment.center),

animate\_rotation=ft.animation.Animation(300,

ft.AnimationCurve.BOUNCE\_OUT),

)

def animate(e):

c.rotate.angle += pi / 2

page.update()

page.vertical\_alignment = ft.MainAxisAlignment.CENTER

page.horizontal\_alignment = ft.CrossAxisAlignment.CENTER

page.spacing = 30

page.add(

c,

ft.ElevatedButton("Animate!", on\_click=animate), #

Анимировать!

)

ft.app(target=main)

# Анимация масштабирования

Установка параметра элемента управления в

animate\_scale

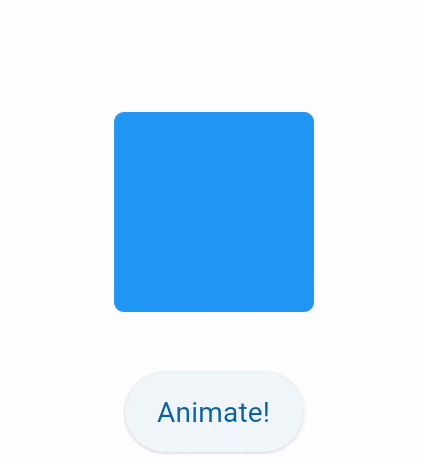
значение , number или экземпляр

True

класса animation.Animation обеспечивает неявную анимацию

свойства .

Control.scale



import flet as ft

def main(page: ft.Page):

c = ft.Container(

width=100,

height=100,

bgcolor="blue",

border\_radius=5,

scale=ft.transform.Scale(scale=1),

animate\_scale=ft.animation.Animation(600,

ft.AnimationCurve.BOUNCE\_OUT),

)

def animate(e):

c.scale = 2

page.update()

page.vertical\_alignment = ft.MainAxisAlignment.CENTER

page.horizontal\_alignment = ft.CrossAxisAlignment.CENTER

page.spacing = 30

page.add(

c,

ft.ElevatedButton("Animate!", on\_click=animate),

)

ft.app(target=main)

# Анимация смещения

Установка параметра

animate\_offset

элемента управления в

значение True , number или экземпляр класса неявно анимировать свойство

|  |  |  |
| --- | --- | --- |
|  | animation.Animation | |
| Control.offset | | . |

позволяет

Свойство - это экземпляр класса transform.Offset , задающий

offset

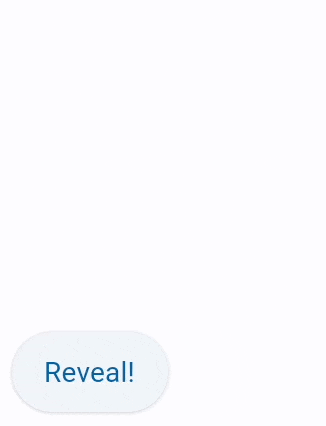
горизонтальное x и вертикальное y смещение элемента управления, масштабированное по его размеру. Например, смещение

transform.Offset(-

приведет к горизонтальному перемещению на четверть ширины элемента управления.

0.25, 0)

Анимация со смещением используется для различных эффектов скольжения:



import flet as ft

def main(page: ft.Page):

c = ft.Container(

width=150,

height=150,

bgcolor="blue",

border\_radius=10,

offset=ft.transform.Offset(-2, 0),

animate\_offset=ft.animation.Animation(1000),

)

def animate(e):

c.offset = ft.transform.Offset(0, 0)

c.update()

page.add(

c,

ft.ElevatedButton("Reveal!", on\_click=animate), # Показать

)

ft.app(target=main)

# Анимация положения

Установка параметра

animate\_position

элемента управления в

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| animation.Animation | | | | | |
| top | , | right | и | bottom | . |

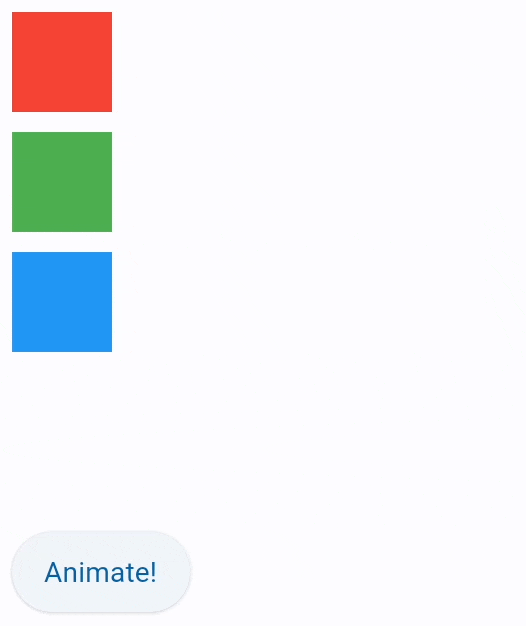
значение True , number или экземпляр класса неявно анимировать свойства контрола left ,

позволяет

Обратите внимание, что положение элемента управления работает только внутри

элемента управления .

Stack



import flet as ft

def main(page: ft.Page):

c1 = ft.Container(width=50, height=50, bgcolor="red",

animate\_position=1000)

c2 = ft.Container(

width=50, height=50, bgcolor="green", top=60, left=0,

animate\_position=500

)

c3 = ft.Container(

width=50, height=50, bgcolor="blue", top=120, left=0,

animate\_position=1000

)

def animate\_container(e):

c1.top = 20

c1.left = 200

c2.top = 100

c2.left = 40

c3.top = 180

c3.left = 100

page.update()

page.add(

ft.Stack([c1, c2, c3], height=250),

ft.ElevatedButton("Animate!", on\_click=animate\_container), #

Анимировать

)

ft.app(target=main)

# Анимированный контейнер

Установка параметра

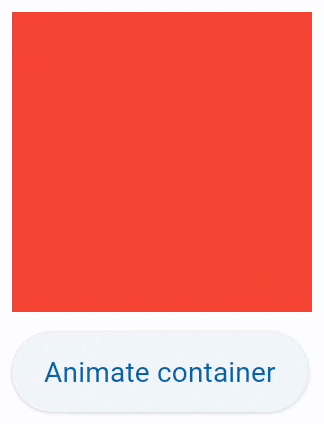
в значение True ,

или

экземпляр класса animation.Animation позволяет неявно анимировать такие свойства контейнера, как размер, цвет фона, стиль границы, градиент.

number

Container.animate



import flet as ft

def main(page: ft.Page):

c = ft.Container(

width=150,

height=150,

bgcolor="red",

animate=ft.animation.Animation(1000,

ft.AnimationCurve.BOUNCE\_OUT),

)

def animate\_container(e):

c.width = 100 if c.width == 150 else 150

c.height = 50 if c.height == 150 else 150

c.bgcolor = "blue" if c.bgcolor == "red" else "red"

c.update()

page.add(c, ft.ElevatedButton("Animate container",

on\_click=animate\_container)) # Анимировать Контейнер

ft.app(target=main)

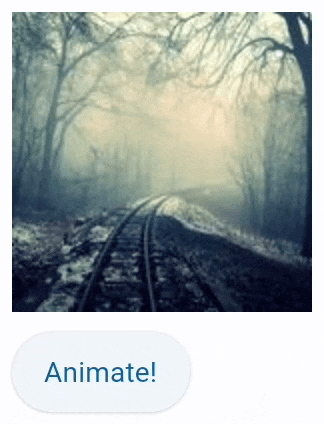
# Анимированный переключатель содержимого

позволяет осуществлять анимированный переход между новым

AnimatedSwitcher

элементом управления и элементом, ранее установленным на в качестве содержимого.

AnimatedSwitcher



import time

import flet as ft

def main(page: ft.Page):

i = ft.Image(src="https://picsum.photos/150/150", width=150,

height=150)

def animate(e):

sw.content = ft.Image(

src=f"https://picsum.photos/150/150?{time.time()}",

width=150, height=150

)

page.update()

sw = ft.AnimatedSwitcher(

i,

transition=ft.AnimatedSwitcherTransition.SCALE,

duration=500,

reverse\_duration=500,

switch\_in\_curve=ft.AnimationCurve.BOUNCE\_OUT,

switch\_out\_curve=ft.AnimationCurve.BOUNCE\_IN,

)

page.add(

sw,

ft.ElevatedButton("Animate!", on\_click=animate), #

Анимировать!

)

ft.app(target=main)

# Обратный вызов (callback) окончания анимации

Все элементы управления со свойствами имеют обработчик

animate\_\*

события on\_animation\_end , который вызывается по завершении анимации и может быть использован для создания цепочки из нескольких анимаций.

Поле данных объекта события содержит название анимации:

* (непрозрачность)

opacity

* (ротация, поворот)

rotation

* (масштаб)

scale

* (смещение)

offset

* position (положение)
* (контейнер)

container

Например:

c = ft.Container(

ft.Text("Анимация!"),

# ...

animate=ft.animation.Animation(1000, "bounceOut"),

on\_animation\_end=lambda e: print("Окончание анимации

контейнера:", e.data)

)

.