

- Simple, clean and consistent design
- If headercolor is not set, it defaults to blue. (Colors can be tweaked in slides\_utils.typ)
- Font family can be changed globally (not per slide) in the slides configuration

```
// Configure the presentation
#show: slides.with(
  ratio: "16-9", // Default is "16-9" if not set
  main-font: "Calibri", // Default value is "Calibri" if not set
  code-font: "Consolas", // Default value is "Consolas" if not set
  font-size-headers: 20pt, // 22pt is the default value if not set
  font-size-content: 19pt, // 20pt is the default value if not set
  footer_text: "", // Text to show in the footer. Empty by default if not set.
  equation_numbering_globally: true, // Default set to "false" if not set.
)
```

**Note:** Example of a gray focusbox with smaller font size to show the global slides configuration. The lightening of the focusbox background color is controlled in slides\_utils.typ with the variable 'percent\_lighter'

You can include images using the `#figure` command:

```
#figure(  
  image("Leonhard_Euler.jpg", width: 15%),  
  caption: [Leonhard Euler],  
) <img:LeonhardEuler>
```



Figure 1: Leonhard Euler

Arguably the GOAT of mathematics: Figure 1 (referenced with Typst's `@` syntax).

**Problem:** A ball is thrown upward with initial velocity  $v_0 = 15 \text{ m/s}$ . Find the maximum height.

Start with the kinematic equation:

$$v^2 = v_0^2 - 2gh \tag{1}$$

At maximum height, the velocity is zero, so we set  $v = 0$ :

$$0 = v_0^2 - 2gh$$

$$h = \frac{v_0^2}{2g}$$

Substitute  $v_0 = 15 \text{ m/s}$  and  $g = 9.8 \text{ m/s}^2$ :

$$h = \frac{15^2}{2 \times 9.8} = 11.5 \text{ m} \tag{2}$$

Only Equation 1 and Equation 2 are numbered.

Here's Taylor's theorem using Typst math syntax (not LaTeX):

**Taylor's theorem:** Let  $k \geq 1$  be an integer and let the function  $f : \mathbb{R} \rightarrow \mathbb{R}$  be  $k$  times differentiable at the point  $a \in \mathbb{R}$ .

Then there exists a function  $h_k : \mathbb{R} \rightarrow \mathbb{R}$  such that

$$f(x) = \sum_{i=0}^k \frac{f^i(a)}{i!} (x - a)^i + h_k(x)(x - a)^k, \tag{1}$$

and  $\lim_{x \rightarrow a} h_k(x) = 0$ .

Equation numbering is default set to false, but can be turned on by setting `equation_numbering_globally: true` in the slides configuration.

Green header slides are recommended for examples and practical applications.

**Example:** Taylor series for  $e^x$  around  $a = 0$ :

Since  $f(x) = e^x$ , all derivatives are  $f^n(x) = e^x$ , and  $f^n(0) = 1$  for all  $n$ .

Therefore, the Taylor series is:

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots$$

This series converges for all  $x \in \mathbb{R}$ .

**Note:** This is a green x-centered focusbox with 90% width, slightly bigger font size and equation numbering is turned off for only this slide using `slide-equation-numbering: false`.

## Python code for calculating $e^x$ using the Taylor series

Here's a Python example that approximates  $e^x$  using its Taylor series. **Note:** The code is shown in a white focusbox for readability and easy font-size control.

```
import math
def exp_taylor(x, N=10):
    """Approximate e^x using the Taylor series sum_{n=0}^N x^n/n!. """
    total = 1.0 # n = 0 term
    term = 1.0
    for n in range(1, N + 1):
        term = term * x / n
        total = term + total
    return total
x, N = 1.0, 10
approx = exp_taylor(x, N)
print(f"N={N}: {approx}")
print("math.exp(1) =", math.exp(1.0))
```

Cyan header slides are recommended for explicit student tasks and exercises.

### Task

Compute the integral:

$$\int x e^x \, dx \quad (1)$$

### Hint

Use integration by parts:

$$\int u \, dv = uv - \int v \, du \quad (2)$$

choose  $u = x$  and  $v' = e^x$ , then  $u' = 1$  and  $v = e^x$ .

By default, slides are:

- Left-aligned horizontally
- Centered vertically

This slide **overrides both to center content horizontally and vertically.**

```
#slide(headercolor: purple, title: "Centering Slide Content", center_x: true,  
center_y: false)[  
  
]
```

Typst can perform calculations directly in the document:

## Output

### Basic arithmetic:

- Addition: 5
- Multiplication: 56
- Division: 25

### Using variables:

- Mass = 10 kg
- Acceleration = 9.8 m/s<sup>2</sup>
- Force = 98 N

### Math functions:

- $\sqrt{15} = 3.872983346207417$
- $2^8 = 256$
- $\sin\left(\frac{\pi}{2}\right) = 1$

## Code

```
#(2 + 3)
#(7 * 8)
#(100 / 4)

#let mass = 10
#let acceleration = 9.8
#let force = mass * acceleration
#mass
#acceleration
#force

#(calc.sqrt(15))
#(calc.pow(2, 8))
#(calc.sin(calc.pi / 2))
```

## Output

**Squares of first 10 numbers:** 1, 4, 9, 16, 25, 36, 49, 64, 81, 100

**Sum of first 100 natural numbers:**

$$\sum_{i=1}^{100} i = 5050 \quad (1)$$

## Code

```
// Squares
#{
  let squares = ()
  for i in range(1, 11) {
    squares.push(i * i)
  }
  squares.map(str).join(", ")
}

// Sum
#{
  let _sum = 0
  for i in range(1, 101) {
    _sum += i
  }
  [$ sum_(i=1)^100 i = #_sum $]
}
```

## Code

### Output

**Fibonacci sequence (first 12 terms):** 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89

**Powers of 2:** 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024

```
// Fibonacci
#{
  let fib = (0, 1)
  for i in range(2, 12) {
    fib.push(fib.at(-1) + fib.at(-2))
  }
  fib.map(str).join(", ")
}

// Powers of 2
#{
  let powers = ()
  for i in range(0, 11) {
    powers.push(calc.pow(2, i))
  }
  powers.map(str).join(", ")
}
```

This is the first line.

This is the first line.

This is the second line (appears after pause).

This is the first line.

This is the second line (appears after pause).

This is the third line (appears after second pause).

This is the first line.

This is the second line (appears after pause).

This is the third line (appears after second pause).

This is the fourth line. (appears after third pause).

Content 1

Content 1

Content 2

Content 1

Content 2

Content 3

Here are the key points:

- First point is always visible

Here are the key points:

- First point is always visible
- Second point appears after first pause

Here are the key points:

- First point is always visible
- Second point appears after first pause
- Third point appears after second pause

Here are the key points:

- First point is always visible
- Second point appears after first pause
- Third point appears after second pause
- Fourth point appears last

First section starts here.

Second section starts (appears with first section).

First section starts here.

First section continues.

Second section starts (appears with first section).

Second section continues.

First section starts here.

First section continues.

Second section starts (appears with first section).

Second section continues.

### **Introduction**

Some introductory text.

### **Introduction**

Some introductory text.

### **Main Content**

This is important content in a focusbox.

### **Introduction**

Some introductory text.

### **Main Content**

This is important content in a focusbox.

### **Conclusion**

Final thoughts and summary.

This slide has no pauses.

All content appears at once.

This is the expected behavior for backward compatibility.

Let's look at some code:

Let's look at some code:

```
def hello():  
    print("Hello, World!")
```

Let's look at some code:

```
def hello():  
    print("Hello, World!")
```

This function prints a greeting message.