Highlights.jl

Syntax Highlighting Demo

This PDF was generated using Highlights.jl with Typst output. Each section shows the same code in different themes and languages.

Dracula

Julia

```julia
# Fibonacci sequence
function fib(n::Int)
    n <= 1 && return n
    return fib(n - 1) + fib(n - 2)
end

println("fib(10) = $(fib(10))")
```

Python

```python
# Fibonacci sequence
def fib(n):
    if n <= 1:
        return n
    return fib(n - 1) + fib(n - 2)

print(f"fib(10) = {fib(10)}")
```

Rust

```rust
// Fibonacci sequence
fn fib(n: u32) -> u32 {
    if n <= 1 { return n; }
    fib(n - 1) + fib(n - 2)
}

fn main() {
    println!("fib(10) = {}", fib(10));
}
```

Nord

Julia

```julia
# Fibonacci sequence
function fib(n::Int)
    n <= 1 && return n
    return fib(n - 1) + fib(n - 2)
end

println("fib(10) = $(fib(10))")
```

Python

```python
# Fibonacci sequence
def fib(n):
    if n <= 1:
        return n
    return fib(n - 1) + fib(n - 2)

print(f"fib(10) = {fib(10)}")
```

Rust

```rust
// Fibonacci sequence
fn fib(n: u32) -> u32 {
    if n <= 1 { return n; }
    fib(n - 1) + fib(n - 2)
}

fn main() {
    println!("fib(10) = {}", fib(10));
}
```

Solarized Light

Julia

```julia
# Fibonacci sequence
function fib(n::Int)
    n <= 1 && return n
    return fib(n - 1) + fib(n - 2)
end

println("fib(10) = $(fib(10))")
```

Python

```python
# Fibonacci sequence
def fib(n):
    if n <= 1:
        return n
    return fib(n - 1) + fib(n - 2)

print(f"fib(10) = {fib(10)}")
```

Rust

```rust
// Fibonacci sequence
fn fib(n: u32) -> u32 {
    if n <= 1 { return n; }
    fib(n - 1) + fib(n - 2)
}

fn main() {
    println!("fib(10) = {}", fib(10));
}
```

Gruvbox Dark

Julia

```julia
# Fibonacci sequence
function fib(n::Int)
    n <= 1 && return n
    return fib(n - 1) + fib(n - 2)
end

println("fib(10) = $(fib(10))")
```

Python

```python
# Fibonacci sequence
def fib(n):
    if n <= 1:
        return n
    return fib(n - 1) + fib(n - 2)

print(f"fib(10) = {fib(10)}")
```

Rust

```rust
// Fibonacci sequence
fn fib(n: u32) -> u32 {
    if n <= 1 { return n; }
    fib(n - 1) + fib(n - 2)
}

fn main() {
    println!("fib(10) = {}", fib(10));
}
```