Programming for Everybody

2. Control Flow

le wagon

Control Flow

Control flow allows us to select different outcomes depending on a condition

Ruby’s **if statements** evaluate an **expression** to either **true** or **false**

If that expression is true, Ruby executes the block of code that follows the if

If it’s not true, Ruby doesn’t execute that block of code and goes on to the next thing

Control Flow

```
if weather == “sunny”
   puts “wear sunglasses”
end
```

= VS ==

She’s 12 years old → age = 12

Is she 12 years old? → age == 12

If, else, elsif

IF
To evaluate one condition only

ELSE
“otherwise” == run this code if no condition before was true

ELSIF
When we want to have more than two options

If, else, elsif

```
if condition
  # code executed only when condition is "truthy"
elsif another condition
  # code executed only when another condition is “truthy"
else
  # code executed only when all other conditions are not "truthy"
end
```

```
if weather == “sunny"
  puts “Wear sunglasses!”
elsif weather == “rainy"
  puts “Take an umbrella!”
else
  puts “No special equipment needed”
end
```

Unless

Used to check if a condition is **false**

Ex: you don’t want something to happen **unless** a certain condition is met

```
unless weather == “sunny”
   puts “you don’t need sunglasses”
end
```

Relational operators
(comparisons)

==	equal (different from =)
!=	not equal
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to

Bolean operators
(multiple conditions)

Always return **true** or **false**

Check if two conditions are true or false **simultaneously**

AND (&&)

Evaluates to true when **all** expressions are true

OR (II)

Evaluates to true when **at least one** of the expressions is true

Bolean operators
(multiple conditions)

```
if (hour > 9 && hour < 12) || (hour > 14 && hour < 18)
  puts “Time to work!”
else
  puts “Siesta! Take a nap”
end
```

Thank you!