Rust Memory usage

```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```

Thread stack

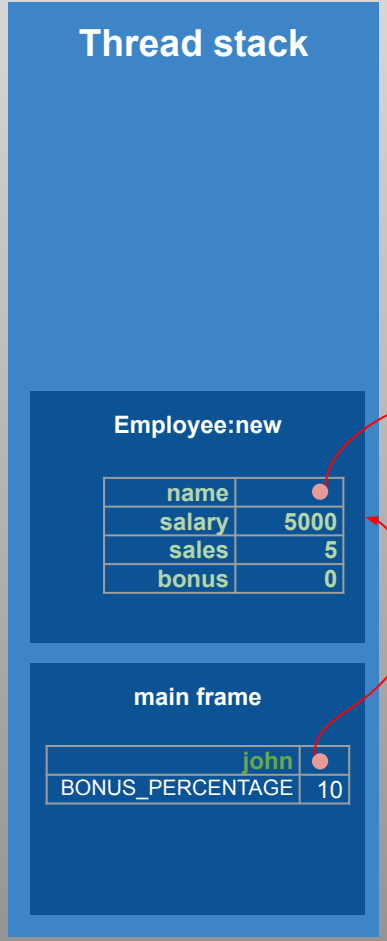
main frame

BONUS_PERCENTAGE	10

Heap

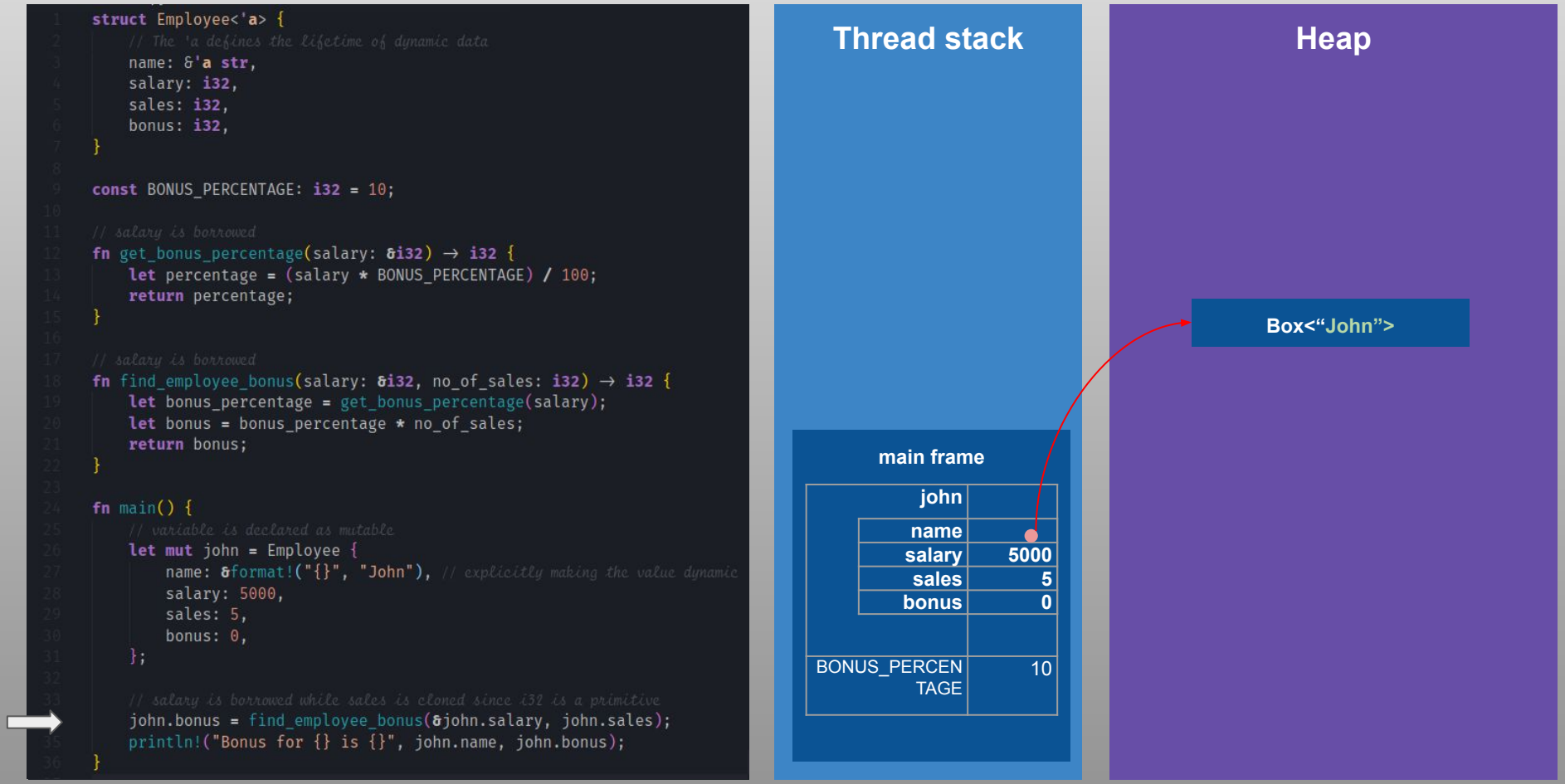
Rust Memory usage

```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```



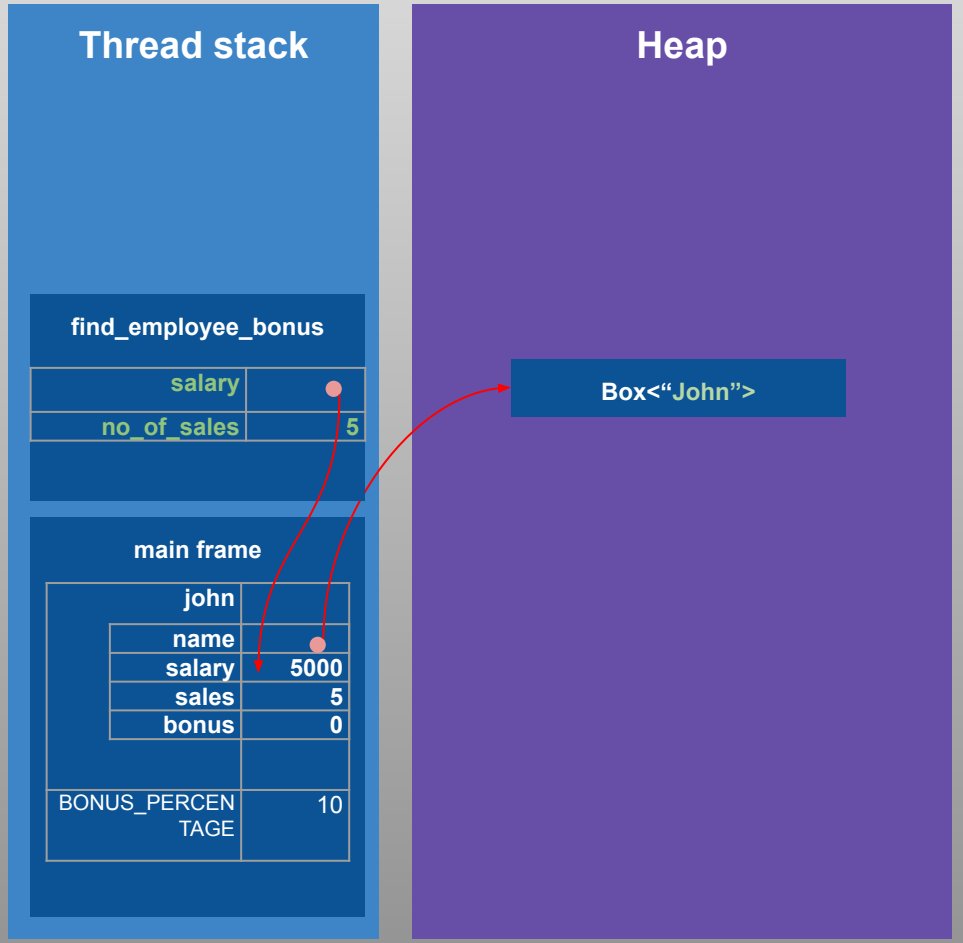
Rust Memory usage

```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```



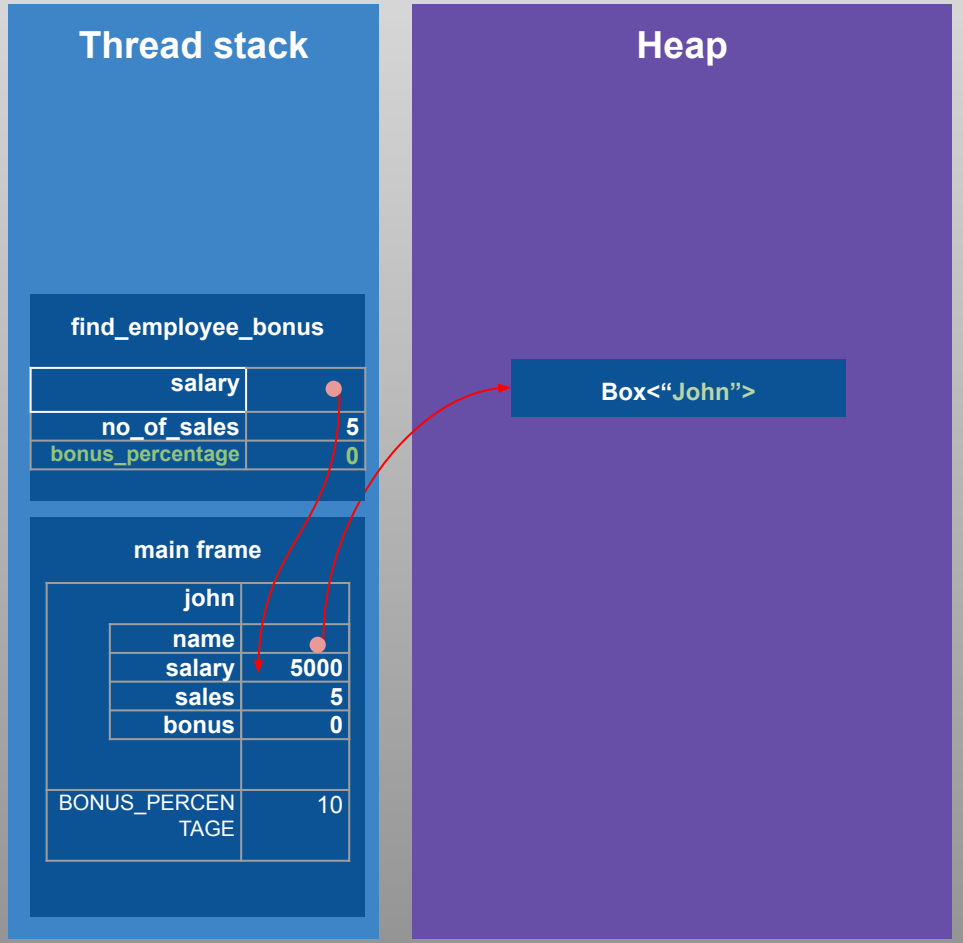
Rust Memory usage

```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```



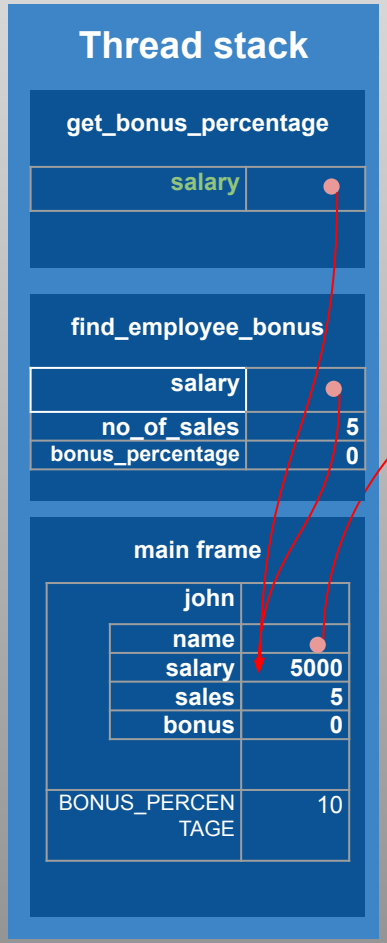
Rust Memory usage

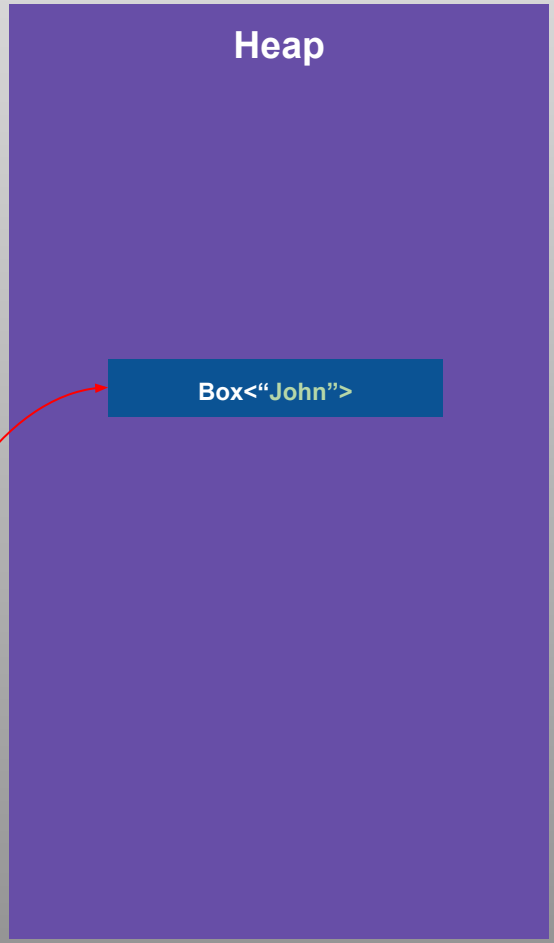
```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```



Rust Memory usage

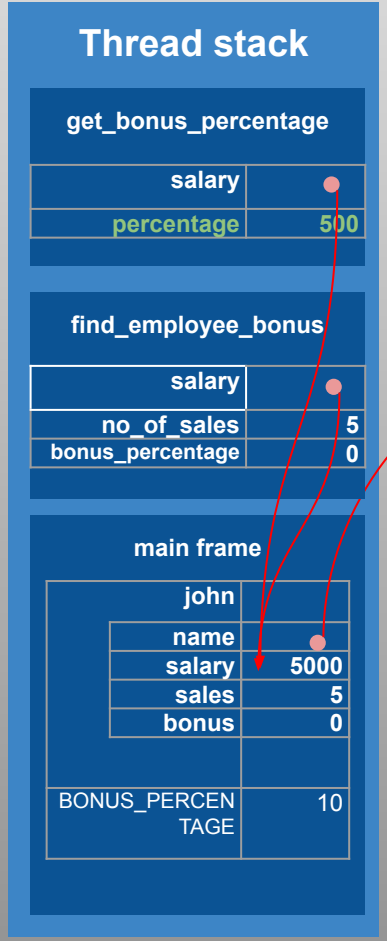
```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```

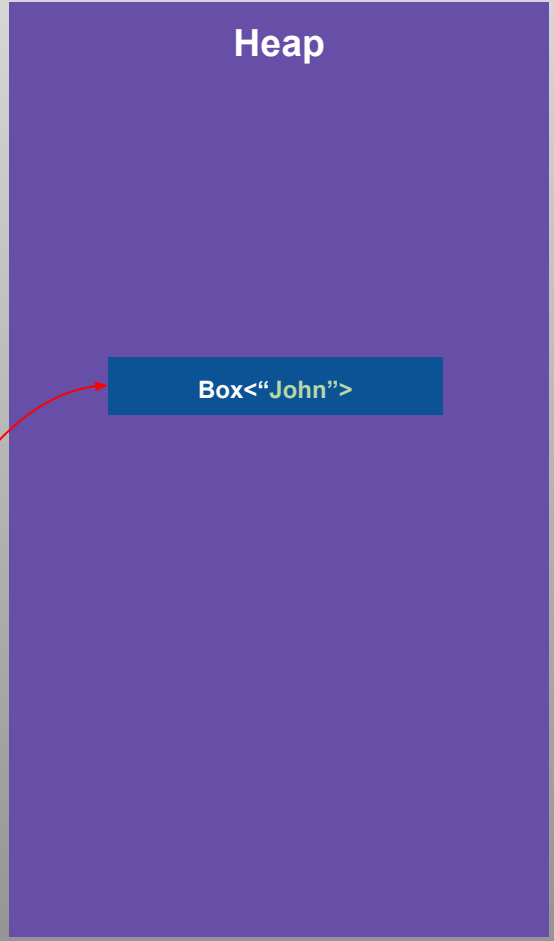




Rust Memory usage

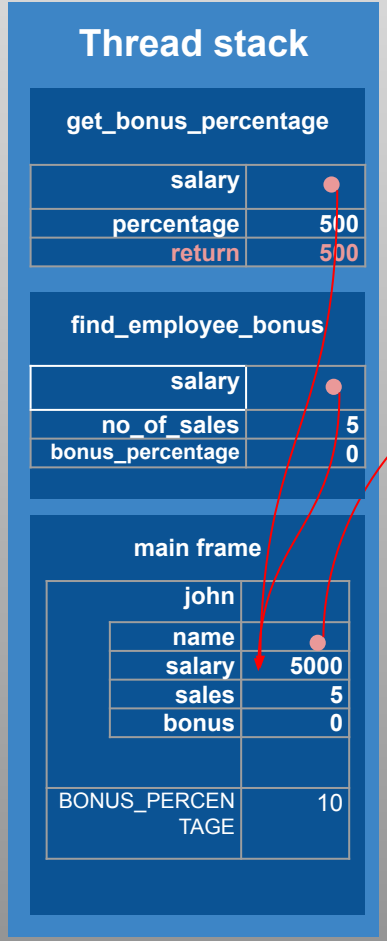
```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```

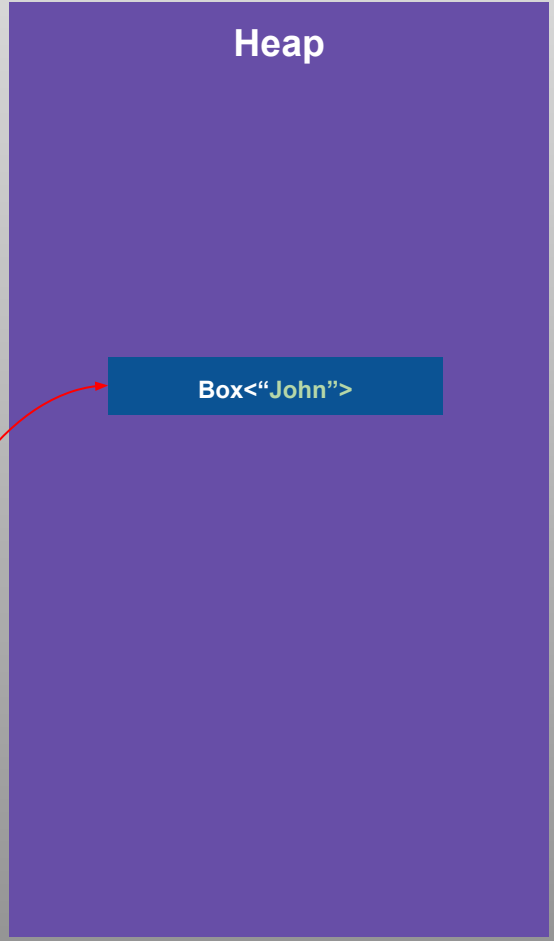




Rust Memory usage

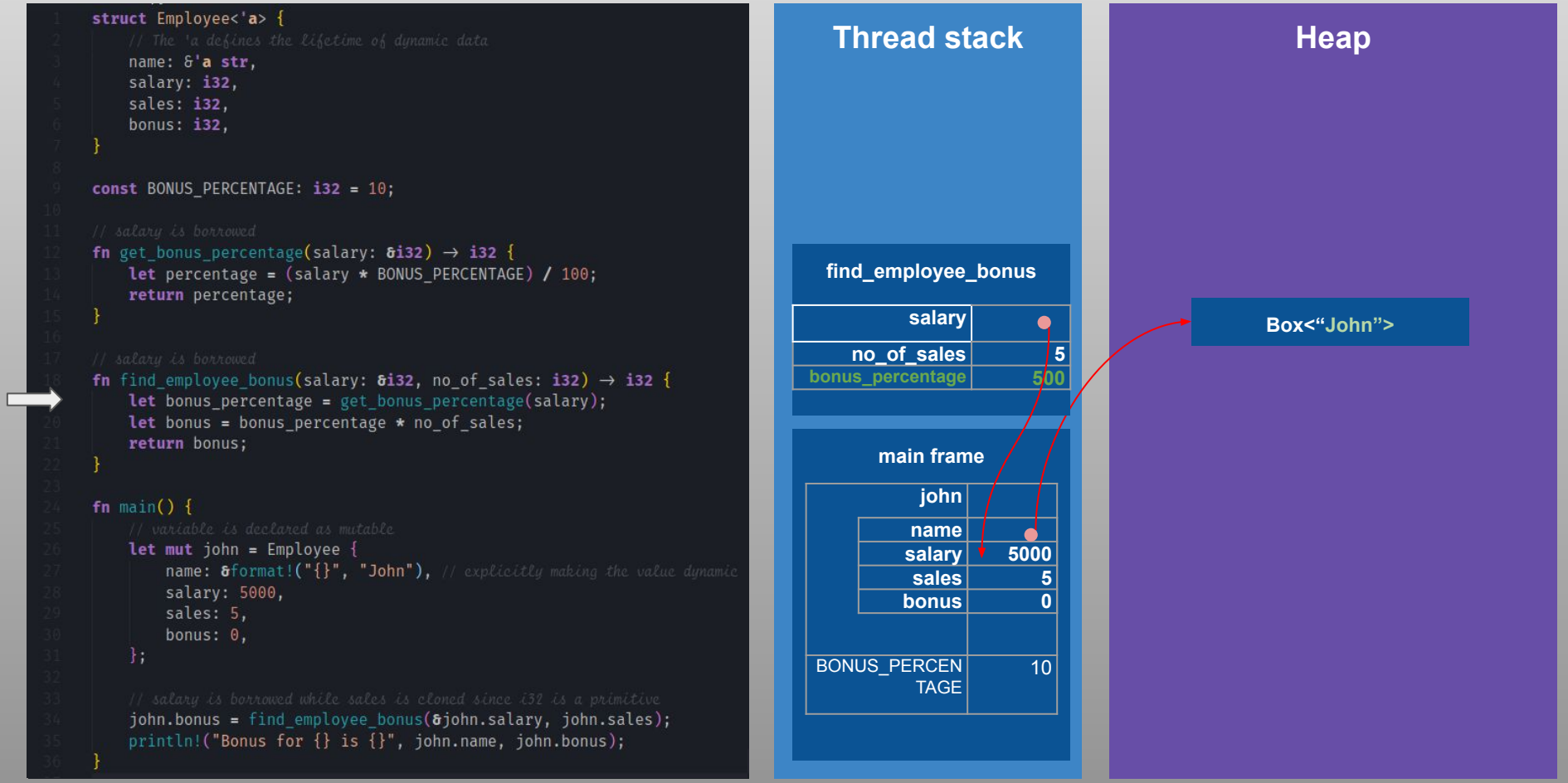
```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```





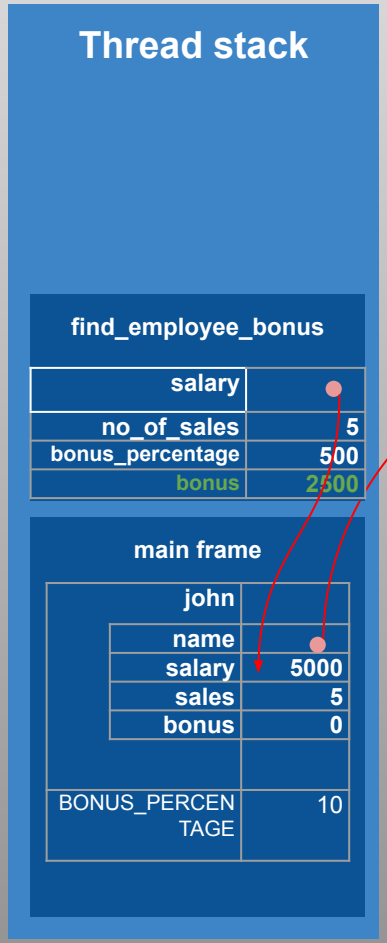
Rust Memory usage

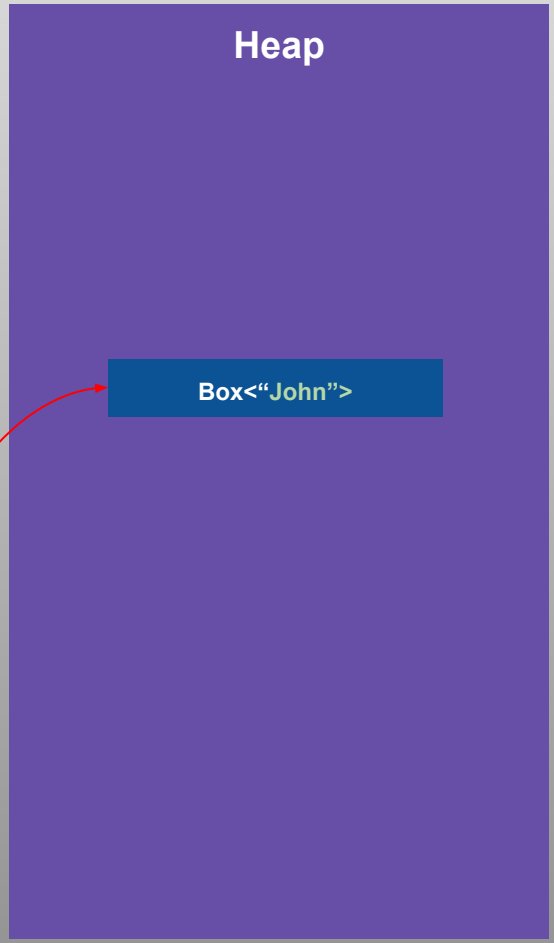
```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```



Rust Memory usage

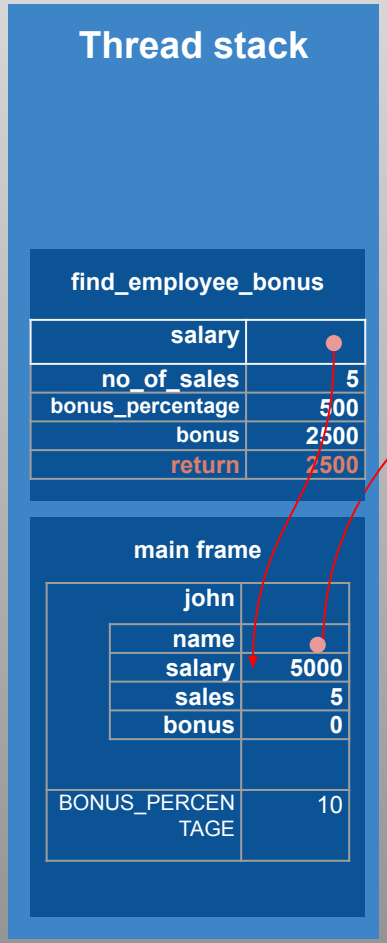
```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```

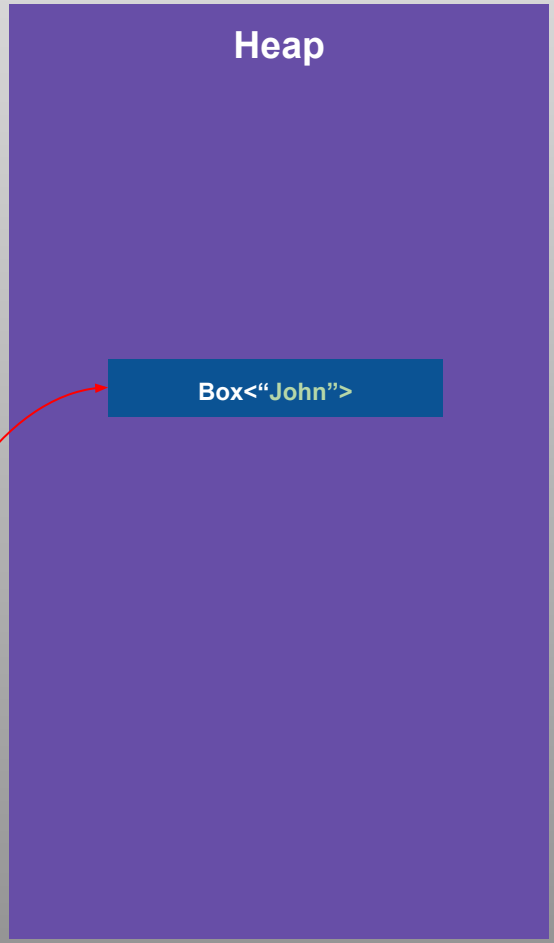




Rust Memory usage

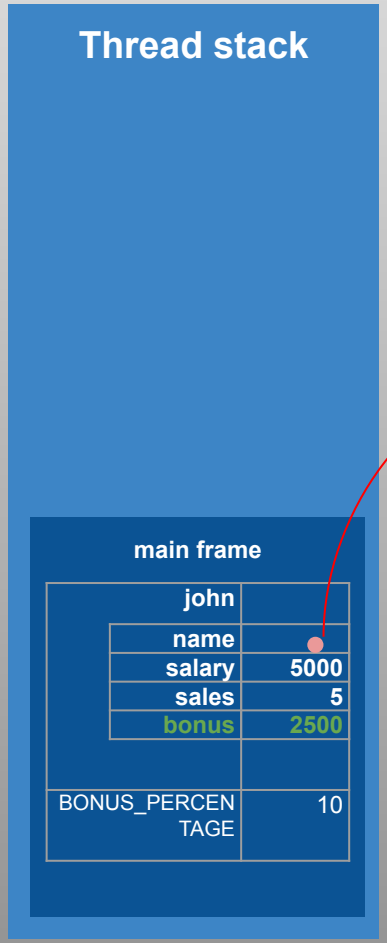
```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```

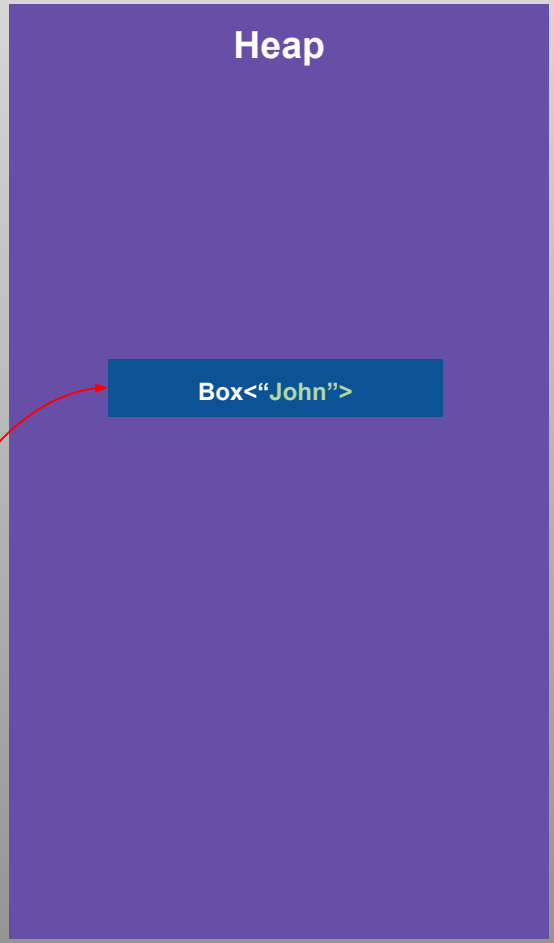




Rust Memory usage

```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```





Rust Memory usage

```rust
struct Employee<'a> {
    // The 'a defines the lifetime of dynamic data
    name: &'a str,
    salary: i32,
    sales: i32,
    bonus: i32,
}

const BONUS_PERCENTAGE: i32 = 10;

// salary is borrowed
fn get_bonus_percentage(salary: &i32) -> i32 {
    let percentage = (salary * BONUS_PERCENTAGE) / 100;
    return percentage;
}

// salary is borrowed
fn find_employee_bonus(salary: &i32, no_of_sales: i32) -> i32 {
    let bonus_percentage = get_bonus_percentage(salary);
    let bonus = bonus_percentage * no_of_sales;
    return bonus;
}

fn main() {
    // variable is declared as mutable
    let mut john = Employee {
        name: &format!("{}", "John"), // explicitly making the value dynamic
        salary: 5000,
        sales: 5,
        bonus: 0,
    };

    // salary is borrowed while sales is cloned since i32 is a primitive
    john.bonus = find_employee_bonus(&john.salary, john.sales);
    println!("Bonus for {} is {}", john.name, john.bonus);
}
```