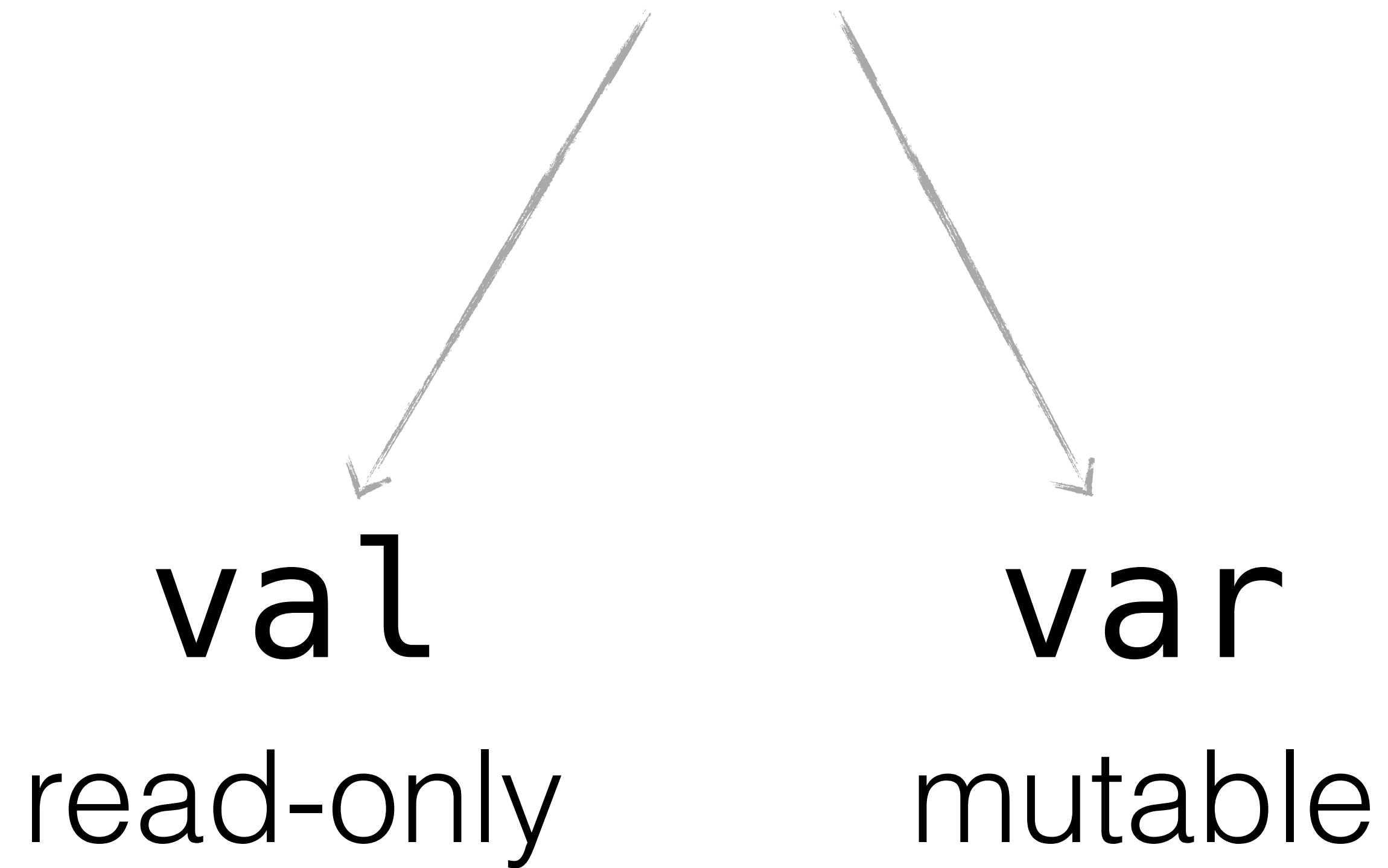




Variables

Variables



Read-only variable: `val`

```
val question: String =  
    "life, the universe, " +  
    "and everything"  
println("$question")
```

life, the universe, and everything?

Read-only variable: `val`

```
val question: String =  
    "life, the universe, " +  
    "and everything"  
println("$question")
```

```
question = "sure?"
```

Compiler error:
`val` cannot be reassigned

corresponds to a `final` variable in Java

Mutable variable: `var`

```
var answer = 0  
answer = 42  
println(answer)           // 42
```

Local type inference

val greeting = "Hi!"
: String

var number = 0
: Int

String and Int types are inferred



Why doesn't the following code compile?

```
var string = 1  
string = "abc"
```

Because you can't

1. re-assign var
2. assign an integer value to a variable of `String` type
3. assign a string literal to a variable of `Int` type





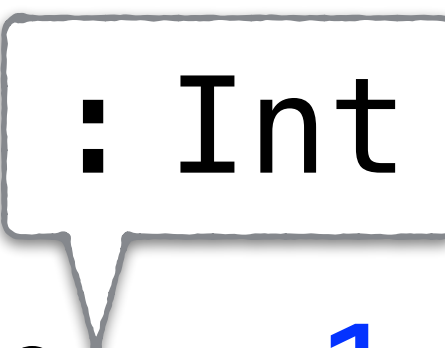
Why doesn't the following code compile?

```
var string = 1  
string = "abc"
```

Because you can't

1. re-assign var
2. assign an integer value to a variable of String type
3. assign a string literal to a variable of Int type

Local type inference



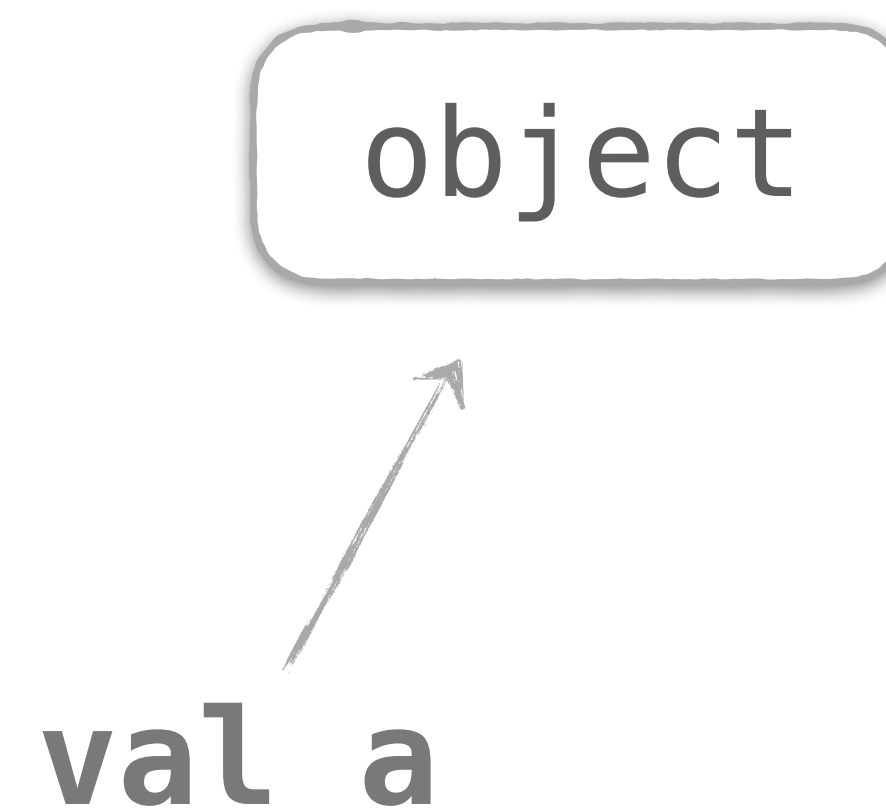
```
var string = 1  
string = "abc"
```

Compiler error: Type mismatch:
inferred type is String
but Int was expected



Is it possible to modify
an object stored in `val`?

1. yes
2. no







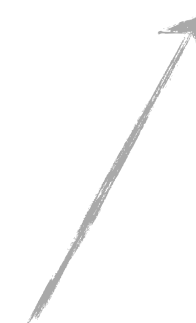
Is it possible to modify
an object stored in `val`?

1. yes

2. no

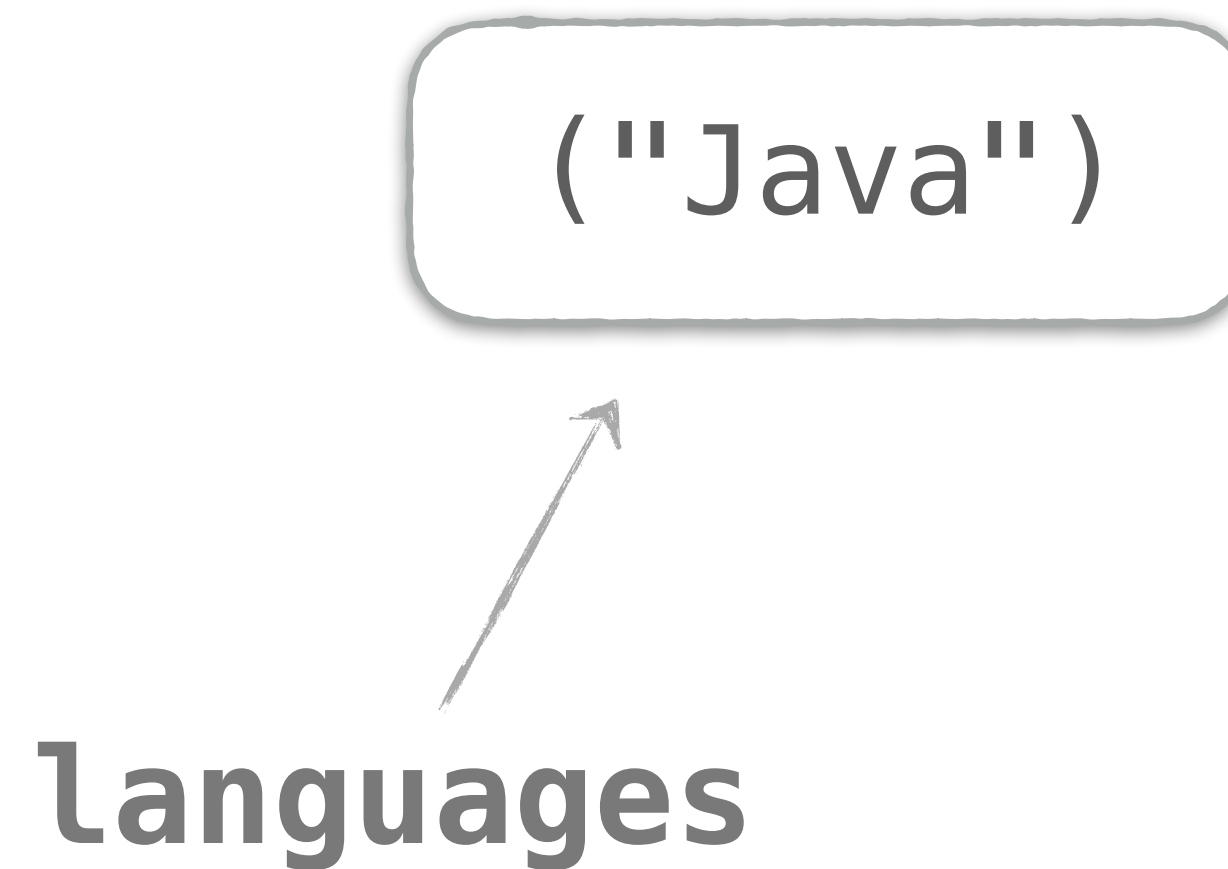
object

`val a`



val: read-only reference, not object

```
val languages = mutableListOf("Java")
```



`val`: read-only reference, not object

```
val languages = mutableListOf("Java")  
languages.add("Kotlin")
```



The diagram illustrates the state of the `languages` variable. It is represented by a rounded rectangular box containing the text `("Java", "Kotlin")`. An arrow points from the variable name `languages` below to this box, indicating that the variable holds a reference to this list object.

```
graph BT; languages --> list["("Java", "Kotlin")"]
```

`("Java", "Kotlin")`

`languages`



Why doesn't the following code compile?

```
val list = listOf("Java")  
list.add("Kotlin")
```

Because you can't

1. modify a read-only list
2. modify an object stored in `val`





Why doesn't the following code compile?

```
val list = listOf("Java")  
list.add("Kotlin")
```

Because you can't

1. modify a read-only list

2. modify an object stored in `val`

Lists: mutable & read-only

```
val mutableList = mutableListOf("Java")  
mutableList.add("Kotlin")
```

```
val readOnlyList = listOf("Java")  
readOnlyList.add("Kotlin")
```

Read-only list lacks mutating methods

Prefers `vals` to `vars`

Don't omit types (specify them explicitly)
if they might be not clear from the context