**Declaring Variables in Go**

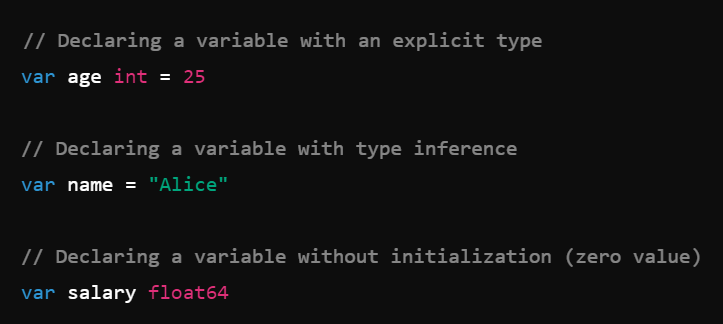
**Variable Declaration Syntax**

In Go, variables can be declared using the `**var**` keyword or the short variable declaration `**:=**`. Here’s a guide on how to declare variables:

**1. Using the `var` Keyword:**

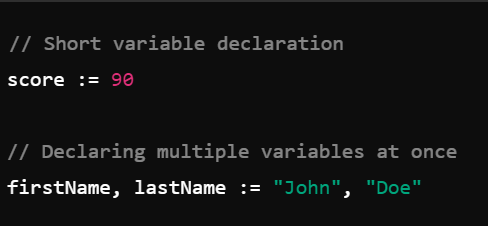
* The `**var**` keyword is used for declaring a variable. You can optionally specify the type of the variable.
* If you don't specify the type, Go will infer it from the initial value.
* Syntax: **`var variableName type = initialValue`**

Examples:



**2. Using Short Variable Declaration:**

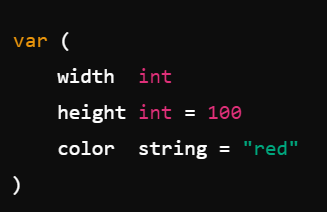
* The short variable declaration uses **`:=`** and is a shorthand for declaring and initializing variables.
* This can only be used within functions.
* Syntax**: `variableName := initialValue`**



**3. Declaring Multiple Variables:**

* You can declare multiple variables at once using the `**var**` keyword with a block syntax.
* This is useful for grouping related variable declarations together.

Example:



**Zero Values**

When a variable is declared without an initial value, it is assigned a zero value by default:

- `**int**`, `**float64**`, etc.: `**0**`

- `**string**`: **""**(empty string)

- `**bool**`: `**false**`

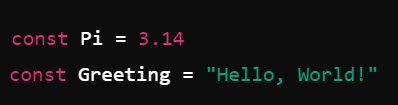
- **Pointers, slices, maps, channels, interfaces**: `**nil**`

**Constants**

Constants are declared using the `**const**` keyword and must be assigned a value at the time of declaration. They cannot be changed later.

- Syntax: `const constantName type = value`

- Example:



**Summary**

- Use `**var**` for variable declarations, with optional type specification.

- Use `**:=**` for short variable declarations within functions.

- Variables without an initial value get a zero value.

- Constants are declared with `**const**` and cannot be modified.

These variable declaration techniques allow Go to be both expressive and efficient, accommodating different coding styles and needs.