📘 Default Parameters in C++

In C++, **default parameters** allow you to specify **default values** for function arguments. This enables functions to be called with **fewer arguments** than they are defined to accept, improving flexibility and readability.

🔹 Syntax of Default Parameters

A default parameter is specified by assigning a value in the **function declaration**:

```
void greet(std::string name = "Guest");
```

This allows the function to be called with or without an argument:

```
greet();            // Uses default: "Guest"
greet("Alice");     // Uses provided argument
```

🔸 Default Parameters in Member Functions

When using default parameters in **class member functions**, the default values should be specified ***only in the declaration***, not in the definition.

✅ Correct Usage

```
// Header file or class definition
class Greeter {
public:
    void greet(std::string name = "Guest");  // Default specified here
};

// Source file
void Greeter::greet(std::string name) {
    std::cout << "Hello, " << name << "!" << std::endl;
}
```

❌ Incorrect Usage

```
// Header file
class Greeter {
public:
    void greet(std::string name);  // No default here
};

// Source file
void Greeter::greet(std::string name = "Guest") {  // ❌ Error!
    std::cout << "Hello, " << name << "!" << std::endl;
}
```

Why is this incorrect?
C++ allows default arguments to be specified **only once**—typically in the declaration. Specifying them again in the definition leads to a **compiler error** or **ambiguity**.

🧠 Key Rules and Best Practices

- ✅ **Specify default arguments in the function declaration**, especially in header files.
- ❌ **Do not repeat default arguments in the function definition.**
- ❌ Default arguments must appear **at the end** of the parameter list. If there are any required parameters, they must appear before the default parameters in the parameter list.
- Use default parameters to simplify overloaded functions when the behavior is similar.
- Avoid using default parameters in virtual functions if polymorphism is involved, as default arguments are **statically bound**.

🧪 Example: Class with Default Parameters

```
#include <iostream>
#include <string>

class Logger {
public:
    void log(std::string message, std::string level = "INFO");
};

void Logger::log(std::string message, std::string level) {
    std::cout << "[" << level << "] " << message << std::endl;
}

int main() {
    Logger logger;
    logger.log("System started");                // Uses default level "INFO"
    logger.log("Disk full", "WARNING");          // Uses provided level
    return 0;
}
```

🎯 Quiz!

Here's a short quiz on the topic: quiz

Footer Separator

🔭 Markdown Viewer

How to view the markdown files in a browser...

- Markdown Viewer

🧠 Lecture Practices

Here are the lecture Practices...

- Day 1
- Day 2
- Day 3

🔍 Lecture Quizzes

Here are the lecture quizzes...

- Day 1
- Day 2
- Day 3

⚙️ Weekly Topics

Here are the topics for the week...

- Calling Methods
- Calling Methods 2
- Creating Methods
- Iterators
- Vectors
- References
- Const
- Erasing in a Loop
- Default Parameters