**Source:** [C++ style examples](https://www.google.com/search?q=C%2B%2B+style+examples&sca_esv=6fe806cc0ee527f3&sxsrf=AHTn8zo0XsUrDWxW07wo9RuiTFWvbGyIFA%3A1739906604709&source=hp&ei=LN60Z-GqKKbB0PEPo9SLgA8&iflsig=ACkRmUkAAAAAZ7TsPP64zWaOjWqDH2XiVB43mDdYfQ2Q&ved=0ahUKEwjh8POY-c2LAxWmIDQIHSPqAvAQ4dUDCBk&uact=5&oq=C%2B%2B+style+examples&gs_lp=Egdnd3Mtd2l6IhJDKysgc3R5bGUgZXhhbXBsZXMyBhAAGBYYHjILEAAYgAQYhgMYigUyCBAAGIAEGKIEMgUQABjvBTIIEAAYgAQYogQyCBAAGIAEGKIEMggQABiiBBiJBUicKFAAWLUmcAB4AJABAJgBfKAB9AuqAQQxMy41uAEDyAEA-AEBmAISoAKnDMICChAjGIAEGCcYigXCAgsQABiABBiRAhiKBcICDhAuGIAEGLEDGNEDGMcBwgIOEAAYgAQYsQMYgwEYigXCAggQABiABBixA8ICERAuGIAEGLEDGNEDGIMBGMcBwgILEAAYgAQYsQMYgwHCAgUQABiABJgDAJIHBDEzLjWgB4Fv&sclient=gws-wiz) – Google Search (2/18/25)

C++ style conventions enhance code readability and maintainability. Here are some common examples:

**Naming Conventions**

* **CamelCase**: for class names (e.g., MyClass).
* **snake\_case**: for function and variable names (e.g., calculate\_area).
* **ALL\_CAPS**: for constants (e.g., MAX\_VALUE).
* **m\_prefix**: for member variables (e.g., m\_name).

**Indentation and Spacing**

* Use 2 or 4 spaces for indentation; avoid tabs.
* Keep lines within 80-120 characters.
* Add a blank line between logical code blocks.

**Comments**

* Use // for single-line comments.
* Use /\* ... \*/ for multi-line comments when necessary.
* Write clear, concise comments explaining the purpose of code sections.

**File Structure**

* Header files (.h or .hpp) for declarations.
* Source files (.cpp) for definitions.
* Include headers in a logical order, often grouped by project, library, and system includes.

**Example**

#include <iostream> // System header  
#include "my\_class.h" // Project header  
  
const int MAX\_SIZE = 100;  
  
class MyClass {  
public:  
 MyClass(int value);  
 int get\_value() const;  
 void set\_value(int value);  
  
private:  
 int m\_value;  
};  
  
MyClass::MyClass(int value) : m\_value(value) {}  
  
int MyClass::get\_value() const {  
 return m\_value;  
}  
  
void MyClass::set\_value(int value) {  
 if (value >= 0 && value <= MAX\_SIZE) {  
 m\_value = value;  
 } else {  
 std::cerr << "Error: Value out of range." << std::endl;  
 }  
}  
  
int main() {  
 MyClass obj(50);  
 std::cout << "Value: " << obj.get\_value() << std::endl;  
 obj.set\_value(120); // This will print an error message  
 return 0;  
}