C++ Does Not Name a Type: Get Instant Solutions Inside

by Position is Everything

C++ does not name a type error can occur due to multiple reasons like using an undefined class as a member, initializing the variables at wrong positions, etc. The earlier you figure out the issue, the sooner you'll be able to fix it.

Hence, this post has been curated to let you know about the **potential causes** and provide you with the best fixing procedures. After reading this comprehensive guide, you'll know what's the next step to bring your program back from the trauma of the error.

Contents [show]

Which Reasons Can Cause C++ Does Not Name a Type Error?

The C++ does not name a type error that occurs due to using an undefined class member, undeclared class pointer, or reference, or incorrectly defining the variables. Also, messing up the C++ syntax or using a class without specifying its namespace can put you in trouble and cause the same error.

RECENT POSTS


Modulenotfounderror No Module Named setuptools_rust: Fix It!
By Position is Everything


Command Phasescriptexecution Failed With a Nonzero Exit Code
By Position is Everything

Each problem is discussed below, along with an example to help you find the culprit in your program within minutes.

– Not Defining a Class Before Using It As a Member

If you use a class or struct as a member inside another class or struct before defining it, you'll receive the **struct does not name a type** error. It would be beneficial to know that the compiler needs to calculate the size of a class or a struct when it is defined to know the space that the same class will occupy.

Now, if you add an undefined class member with an unknown size in another class, **the compiler will throw an error**. It indicates that the compiler isn't able to calculate the class's size. An example of this has been attached below:

```
class Animals

{

public:

/* using a class member before defining it */

Petfood catFood;

};

class Petfood

{

public:

/* define the class here */

};
```

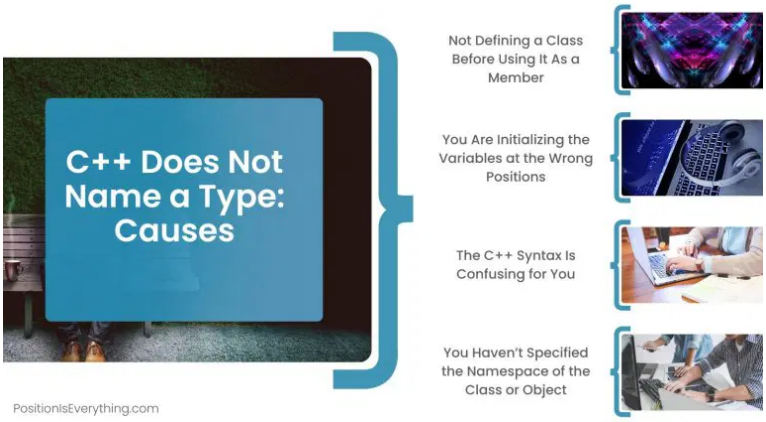
– Using a Class Reference or Pointer Even Before the Class Declaration

Although **using a class reference** or pointer before defining the same class is acceptable, you'll get an error if the class isn't even declared before it.


List Indices Must Be Integers or Slices, Not Tuple: Fixed
By Position is Everything


Unknown Label Type Continuous: Fix Your Scikit-Learn Code
By Position is Everything


Cannot Find Symbol Java Error: Debug and Fix Your Code
By Position is Everything



For example, you haven't declared a class called "Books," and you are using its reference in your "Shelf" class. In this case, the **does not name a type** C++ struct error will show up. Please look at the code below for clarification of the problem:

```
class Shelf

{

public:

void getBooks(Books& mybook);

};
```

– You Are Initializing the Variables at the Wrong Positions

Variable declaration and definition **can be broken down into two steps**. However, if you define a variable in a separate step and that step isn't included inside a function body, you'll be given an error.

Think about it this way: you have created a struct and declared two variables inside it. Next, you try to initialize the variables without wrapping them inside a function. In such a scenario, the "**variable does not name a type – C++**" error will appear on your screen. You can see the code block for the given scenario here:

```
struct Players

{

int football;

int tennis;
```

```
};

Players currentPlayers;

currentPlayers.football = 5;

currentPlayers.tennis = 7;
```

– The C++ Syntax Is Confusing for You

Having confusion in the C++ syntax can lead you to make mistakes in your program and result in the stated error. **The most common mistakes** noticed in even the simple CPP files have been listed below:

- Missing, extra, or improperly placed semicolons and curly brackets.
- Function calls before the main() function.
- Not understanding the difference between various operators.
- Having numbers specified as strings to use them as integers.

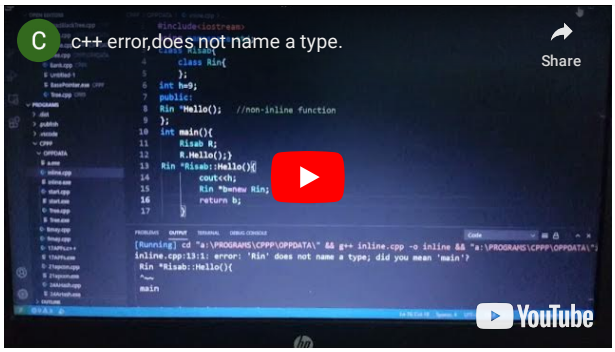
– You Haven't Specified the Namespace of the Class or Object

Having done everything right, if you are getting the same error, there might be a missing namespace. To understand it better, imagine that you are using **the vector class** present in the std namespace.

If you use the class without preceding it with its namespace and the double colon symbol "::," the **vector does not name a type** error will be thrown during program compilation.

Similarly, if you use the cout object without a leading std namespace, the **cout does not name a type in C++** will occur.

How To Make the C++ Does Not Name a Type Error Fly Away?



You can push away the C++ does not name a type error from your screen by being careful with class definitions, **leveraging forward declaration,** or defining the variables correctly. Moreover, following the **C++ syntax** correctly and preceding the class names with their namespaces can save your day.

You can read more about each solution below to identify and implement the best one immediately.

– Define the Class Before Using It As a Member

As you know that it's crucial to define a class or struct before using it as a member inside another class or struct, so it would be better to implement it in your code. So, to remove the **does not name a type C++ class** error received in the previous example, you'll have to swipe the positions of the Animals and Petfood classes.

Here you go with a working code snippet:

```
class Petfood

{

public:

/* define the class here */

};

class Animals

{

public:

Petfood catFood;

};
```

– Leverage Forward Declaration

The forward declaration can help you eliminate the error using an undefined class's reference or pointer inside another class. The term refers to **pre-defining or declaring a class** to meet the flow of your program.

Continuing with the Books class example stated above, it would be better to **forward declare it** before using its reference in the Shelf class. Please refer to the following coding representation to leverage the forward declaration:

```
class Books;

class Shelf

{

public:

void getBooks(Books& mybook);

};
```

– Define the Variables While Declaring Them

It would be **best to define the variables** while declaring them to avoid this error. So, for the struct example shared above, here is the code that won't throw any errors:

```
struct Players

{

int football = 5;

int tennis = 7;

};
```

However, if you aren't willing to **declare and define the variables** in a single step, then go through the next solution.

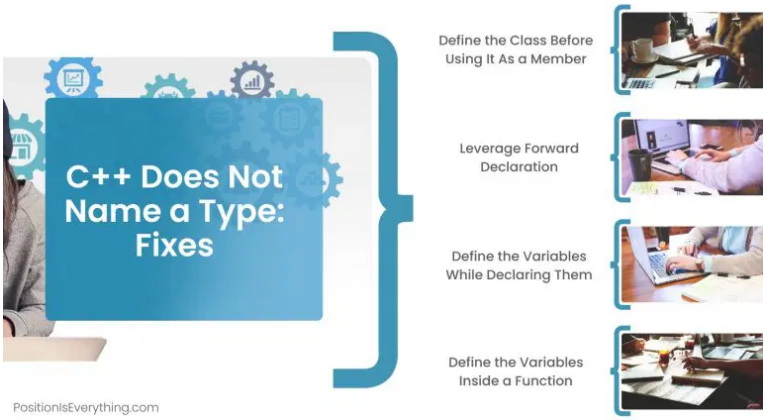
– Define the Variables Inside a Function

Defining the variables inside a function will help you remove the **x does not name a type Arduino** error. Here, x refers to the name of your variable. Please feel free to use the below code block for error-free program compilation.

```
struct Players

{

int football;

int tennis;

};

Players currentPlayers;

void myFunc(){

currentPlayers.football = 5;

currentPlayers.tennis = 7;

}
```

– Precede the Class Name With Its Namespace

It's a good idea to precede the name of the class with its namespace and a double-colon "::" symbol to get rid of the given error. It will ensure that the **compiler identifies the namespace of the class** that you are using and compiles your code without falling for any confusion.



The given way to use a class is often considered better than adding a line like "using namespace xxx" in your file.

So, here you go with the correct way to use the vector class:

```
std::vector
```

– Follow the C++ Syntax Correctly

Writing the correct C++ syntax can save you hours of finding the causes of different errors including the one discussed here and **fixing them**.

So, if you have tried all the solutions and nothing seems to work for you, **double-check the syntax**. Here are a few instructions that you can follow to have a clean and error-free code:

1. Ensure that all statements end with a semicolon.
2. Indent your code properly to stay right with the curly braces.
3. Check and remove any function calls before the main() function.
4. Specify the correct data types for your variables.
5. Replace any inappropriate operators with the correct ones.
6. If you doubt the correctness of some coding statements, accept guidance from Google to help you write them better.

Conclusion

The **stack does not name a type** error might pop up when your compiler is unable to understand your program. Therefore, the causes are often related to mistakes in the syntax or the declaration and definition of different classes, structs, or objects. Here are some statements to help you conclude the post better:

- Define the class prior to using it as a member in another class.
- Go for forward declaration for using class pointers and references.
- Define the variables inside a function or at the time of declaration.
- Precede the class names with their namespaces.
- Double-check your program's syntax.

The more you take care of the details, the better you'll be at **avoiding the error**.

References

- https://stackoverflow.com/questions/2133250/x-does-not-name-a-type-error-in-c
- https://www.codeproject.com/Questions/5265982/Why-does-Cplusplus-say-mynumber-does-not-name-a-ty
- https://forum.arduino.cc/t/compiler-error-does-not-name-a-type/517905/7
- https://discuss.codechef.com/t/does-not-name-a-type-error/96721
- https://stackoverflow.com/questions/8403468/error-vector-does-not-name-a-type
- https://stackoverflow.com/questions/1452721/why-is-using-namespace-std-considered-bad-practice

5/5 - (13 votes)

Author | Recent Posts

Position Is Everything

Your Go-To Resource for Learn & Build: CSS,JavaScript,HTML,PHP,C++ and MYSQL.

Meet The Team

Errors & Warnings

‹ CMake Clean: Is This Command Available for the Cleanup?
› Failed To Set up Listener: SocketException: Address Already in Use

Leave a Comment

Name *

Email *

Website

Save my name, email, and website in this browser for the next time I comment.

POST COMMENT