NAMESPACES & THE SCOPE RESOLUTION OPERATOR

Creating a distinct named scope

Delroy A. Brinkerhoff

FIRST EXAMPLE

```
#include <iostream>
using namespace std;
       .
       .
       .
cout << “hello world” << endl;
```

```
#include <iostream>
       .
       .
       .
std::cout << “hello world”
       << std::endl;
```

NAME COLLISIONS

LIBRARY 1

- `void print(string name) {...}`

LIBRARY 2

- `void print(string part) {...}`

- Programmers name programming elements based on what they do
- A name collision occurs when two elements have the same name in the same scope
- When linked together, elements from multiple object files exist in the same scope

NAMESPACES

- “A namespace is a declarative region that provides a scope to the identifiers (the names of types, functions, variables, etc.) inside it. Namespaces are used to organize code into logical groups and to prevent name collisions that can occur especially when your code base includes multiple libraries”
  - https://learn.microsoft.com/en-us/cpp/cpp/namespaces-cpp
- Namespaces help manage name collision

NAMESPACE CREATES A NAMED SCOPE

```
namespace Acme
{
    void print(string name);
};

namespace Widgets_galore
{
    void print(string part);
};
```

```
void Acme::print(string name)
{
    cout << "Acme " << name << endl;
};

void Widgets_galore::print(string part)
{
    cout << "Widgets_galore "
        << part << endl;
};
```

NAMESPACE CREATES A NAMED SCOPE

```
namespace Acme
{
    void print(string name);
};

namespace Widgets_galore
{
    void print(string part);
};
```

```
void Acme::print(string name)
{
    cout << "Acme " << name << endl;
};

void Widgets_galore::print(string part)
{
    cout << "Widgets_galore "
        << part << endl;
};
```

```
Acme::print("Dilbert");
Widgets_galore::print("wing nut");
```

CLARIFYING AMBIGUOUS SCOPE

- C++ provides functions to convert numbers to strings:
- `string to_string(int val);`
- `string to_string(long val);`
- etc.

```
class beta
{
    public:
        string to_string(int value)
            {return to_string(value);}
};
```

CLARIFYING AMBIGUOUS SCOPE

```
namespace std
{
    string to_string(int value);
};
```

```
class beta
{
    public:
        string to_string(int value)
            {return std::to_string(value);}
};
```