C++ in depth

Initializer

Saurabh Shukla (MySirG)

Agenda

1. Initialization list
2. Initializing fields
3. Initializing const members
4. Initializing reference variables

Initializers

Also known as initializer list.

It allows you which constructor is called and what arguments that constructor receives.

If you have a reference or a const field or if one of the classes used does not have a default constructor, you must use an initialization list

```
class ClassName
{
public:
    ClassName(args): initialization list
    {

    }


};
```

Using initialization list to initialize fields

```
class A
{
    private:
        int a, b, c;

    public:
        A() : a(5), b(6), c(8)
        { }
        A(int a, int b, int c) : a(a), b(b), c(c)
        { }
};
```

fields

Resolving name conflict

Initializing Const member

const member variable can only be initialized using initializer list.

You cannot modify const member variable.

```
class A
{
    private:
        const int K;
    public:
        A()
        {
            K=10;
        }
};
```

Error

```
A(): K(10)
{
}
```

Correct

Initializing reference variable

```
class A
{
    private:
        int &x;
        int K;
    public:
        A() : K(5), x(K)
        { }
};
```

Reference variable

x (*new int)

Where you can use initializer list

1. To initialize reference member variable
2. To initialize const member variable
3. To initialize data members (fields)
4. To invoke base class constructor